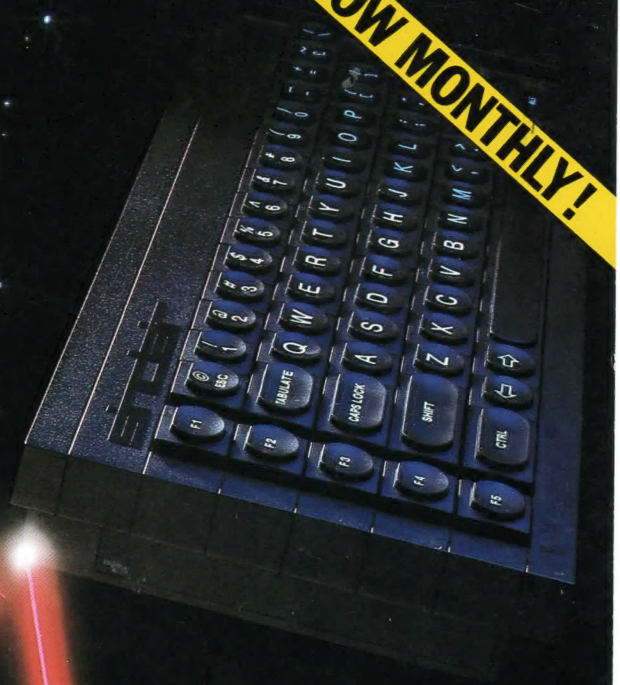
An EMAP Publication

SINCLAIR'S SUPERMICRO

QL USER

February '85 95p

NOW MONTHLY!



SATELLITE STATIONS

QLs in communication

PRINTER IMPRESSIONS

8-page Buyer's Guide of QL Printers

EXCLUSIVE COMPETITION

Win a disk drive, RAM card or executive console

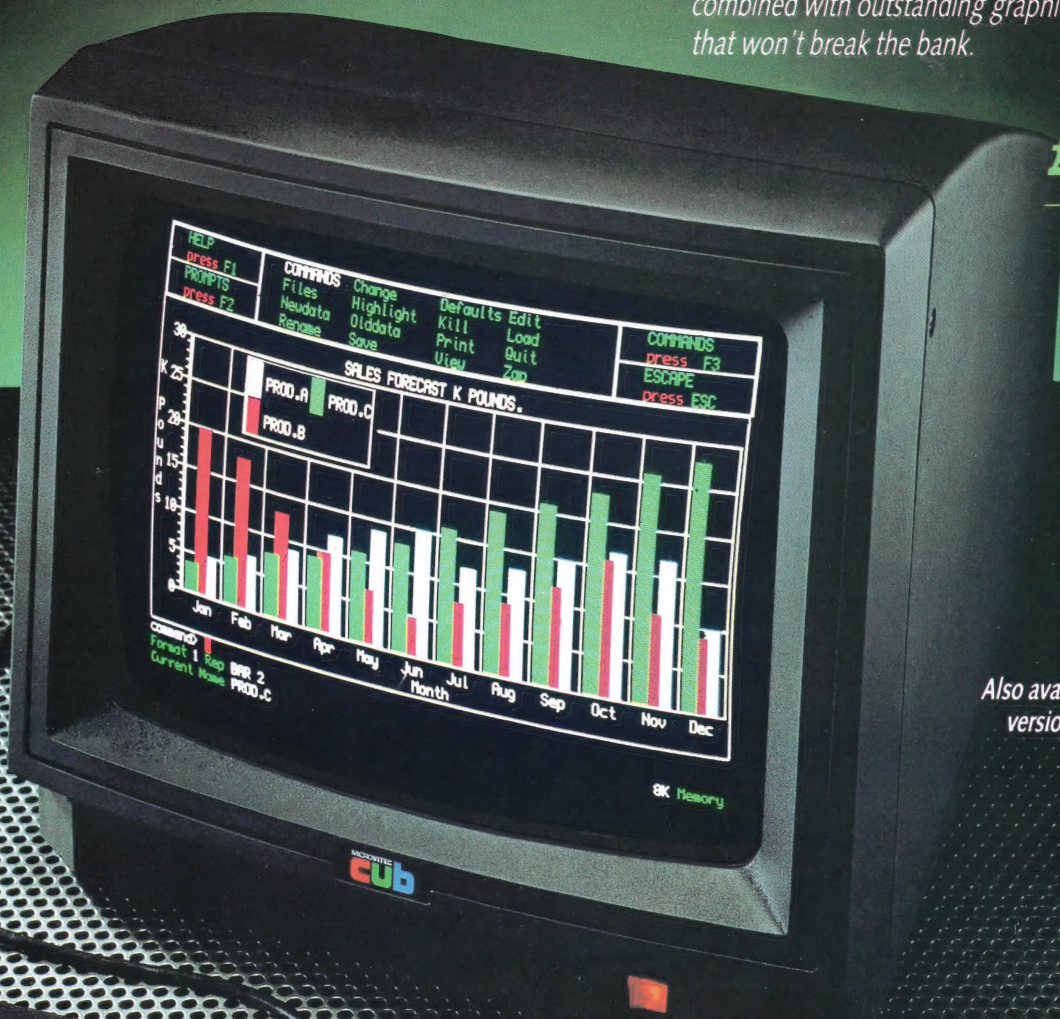
FREE GIFT

AVAILABLE FROM
High Street Computer Retailers
and branches of W. H. Smith, Boots,
John Lewis Partnership, Laskeys, Currys.

EVEN THE PRICE WILL KEEP YOU IN THE BLACK

This sleek, black Microvitec CUB medium resolution colour monitor has been designed to be totally compatible with the Sinclair Q.L. An ability to display 85 column text is combined with outstanding graphic capabilities – at a price that won't break the bank.

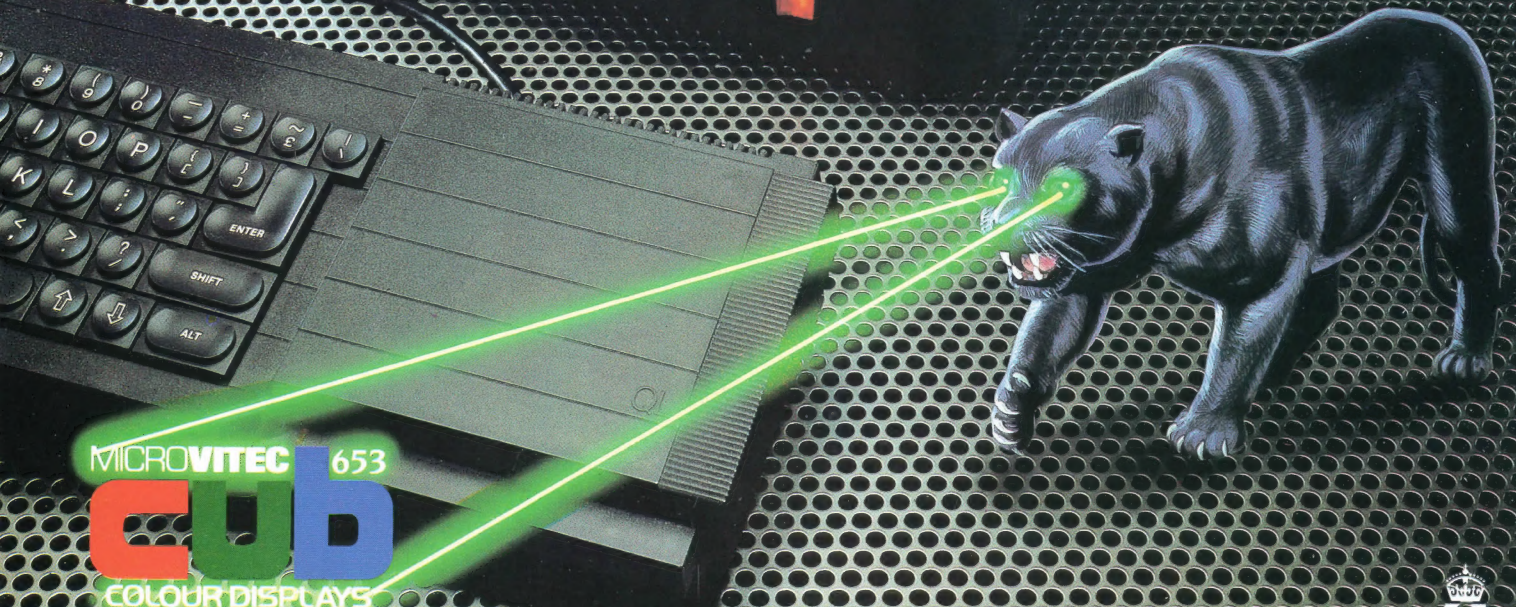
£275
(inc. VAT)



Specification

Model:
CUB 1451/DQ3
14" Q.L. Monitor
RGB TTL input
Tube Resolution (pixels)
653(H) x 585(V)
Dot Pitch 0.43mm
Bandwidth 18 MHz

Also available in Standard Resolution version for only £225 inc. VAT.



MICROVITEC 653
CUB
COLOUR DISPLAYS

Microvitec PLC, Futures Way, Bolling Road, Bradford BD4 7TU, West Yorkshire. Tel: (0274) 390011. Telex: 517717

QL USER

February 1985

Editor	Paul Coster BSc
Technical Editor	Paolo Baccanello
Editorial Assistant	Shirley Eborn
Art Editor	Mike Spiller
Technical Consultant	Adam Denning
Associate Editor	Peter Rodwell
Advertising Manager	Phil Baker
Advert Production	Yvonne Moyser
Sales Executive	Tracey Keighley
Publisher	Alfred Rolington

Contributors

Sid Smith, Leon Heller, Nicky Trevett, Adam Denning, Peter Rodwell, A Carmichael, David Drysdale, A Cryer, Giles Todd, M Newport and P Smith.

Editorial 01-251 6222

QL User, Priory Court,
30-32 Farringdon Lane,
London EC1R 3AU.

Advertising 01-251 6222

Competitions

The winner of the December/January registration draw is Nick Flowers, from Hailsham, East Sussex. He will receive the QL printer interface, donated by Care Electronics, along with our compliments.

Information

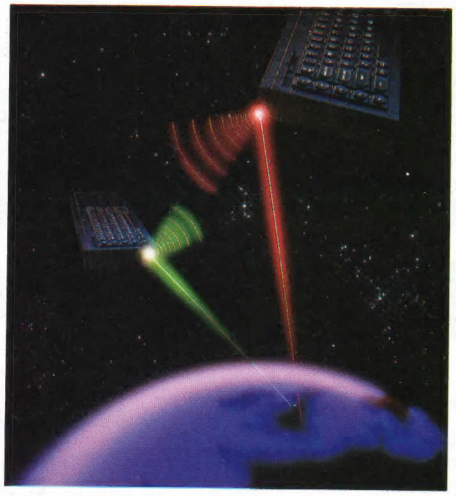
No doubt eagle-eyed readers will have noticed that the photographs in last month's monitor reviews (p.16) were transposed. The Vision QL is shown at the top with Centel's latest underneath – apologies to the two companies concerned.

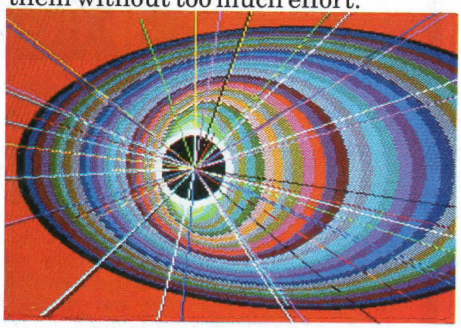
And for all those keen programmers in dismay over the last *two* months because we inadvertently left out the header for our machine code programs; worry no more – it's in this month's selection of readers' programs.

Another slight *faux-pas* concerns last month's Graphic Characters program. First, it was by Leslie Green (whose name was omitted) and second, the arrow on the 'main loop' diagram should go to MOVEQ 2,D0 and not MOVEQ 23,D1.

Finally, a note to all budding authors, whether experienced journalists or keen, first-time writers. We are currently paying top rates (up to £120 per text page) for articles which meet the required standard. All submissions are welcomed and will be carefully considered.

CONTENTS





Published and Distributed by EMAP Business and Computer Publications Ltd.
Typesetting by Contemporary Graphics Ltd., 20 Bowling Green Lane, London EC1.
Printing Riverside Press.

ADD-ONS FOR YOUR QL LOOK NO FURTHER THAN TECHNOMATIC

We've built our reputation on providing a reliable service

- ✱ Wide Range
- ✱ Competitive Prices
- ✱ Deliveries from stock
- ✱ 12 months guarantee on all printers
- ✱ Accessories and consumables readily available

DOT MATRIX PRINTERS

EPSON

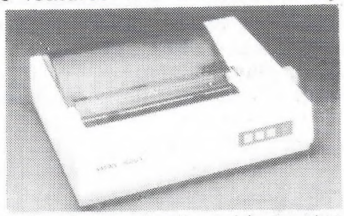
This ever popular range of dot matrix printers set the industry standard and provide reliability and quality 'second to none'. The budget RX80T & RX80FT have 100cps and all standard printing and graphic functions with tractor feed on RX80T and both tractor and friction feed on the RX80FT. The deluxe FX80 has all the above, as well as a 160cps, buffer, programmable characters etc. For wider paper use — upto 15″ — the RX100/FX100 are ideal.

RX80T £215(a)
RX80FT £225(a)
FX100 £315(a)
RX100 £345(a)
FX100 £435(a)

EPSON Serial Interface c/w 2K buffer **£60(c)** (**£52.50** if bought with the printer)
Serial Lead QL/EPSON: 2m long **£12(d)**; 1m long **£8.50(d)**
Spare Ribbon Cartridges: RX/FX 80 **£5.00(d)**
RX/FX 100 **£11.50(d)**
Dust Covers: RX80 **£4(d)** FX80 **£4.50(d)**
Paper Roll Holder **£17(b)**
Tractor Attachment for FX80 **£32(b)**

KAGA

This new range of printers with EPSON compatible control codes feature NEAR LETTER QUALITY print using a 23 x 18 matrix in addition to the features one would normally expect from good quality dot matrix printers. Text modes include Normal, Italic, Enlarged, condensed Super & Sub script and Proportional spacing. Defined characters can be placed in a rom to give personalised print.



An integral 3K buffer, both friction and adjustable tractor feeds with built in paper roll holder, etc, etc make these superb 'value for money' printers unique.

KP810 (80col) £255(a) **KP910 (156col) £359(a)**

Kaga Serial Interface c/w 2K buffer **£85(c)** (**£77.50** if bought with the printer)
Serial Lead QL/Kaga: 2m long **£12(d)**; 1m long **£8.50(d)**
Spare Ribbon Cartridge **£6.75(d)**
Enquire about the special font roms. Phoned 5/3/85

DAISY WHEEL PRINTERS

JUKI 6100 £340(a)

Serial Interface Juki **£65(c)** Tractor Feed Attachment **£99(a)**
Single Sheet Feeder **£199(a)** Ribbon **£2.50(d)**
2K additional RAM for Juki buffer **£7.50(d)**

BROTHER HR15 £340(a)

Serial Interface integral with the printer.

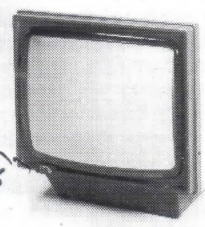
Tractor Feed Attachment **£90(a)** Cut Sheet Feeder **£199(a)**
Ribbon **£4.75(d)**

Plain Fanfold Paper with extra fine perforation (Clean Edge)
2000 Sheets 9.5″ x 11″ **£13(b)**; 2000 Sheets 14.5″ x 11″ **£18.50(b)**
Serial Lead for Daisy Wheels £8.50(d)

SOON TO BE IN STOCK
PARALLEL INTERFACE FOR THE QL £42(b)

MONITORS

The British made MICROVITEC 14″ RGB Monitor with high clarity 85 character display offers a consistent and reliable performance.



MICROVITEC
1451DQ3 £239(a) ~~£274.85~~
Slanting Plinth for Microvitec **£8.50(b)**

KAGA SINCLAIR QL Super Hi Res 12″ RGB Monitor
A high quality monitor with anti glare screen **£273(a)**

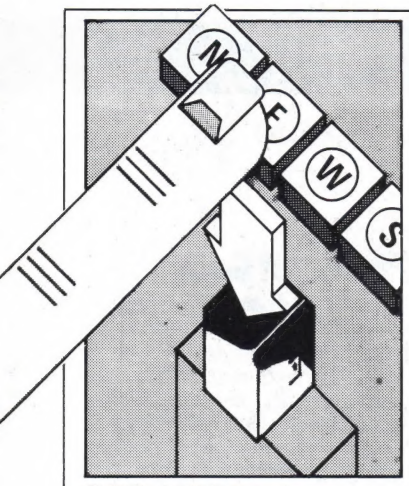
ALL MONITORS SUPPLIED WITH SUITABLE LEADS

TECHNOMATIC LTD

MAIL ORDERS TO: 17 Burnley Road, London NW10 1ED
(Tel: 01-208 1177 Telex 922800)
SHOPS AT: NW London: 15 Burnley Road, London NW10 1ED
⊖ (Dollis Hill 2 mins walk, ample car parking space)
West End: 305 Edgware Road, London W2. Tel: 01-723 0233
(Near Edgware Road) ⊖

01-208 1177

All prices exclude VAT. Please add 50p carriage unless indicated as follows:
(a) £8: (b) £2.50: (c) £1.50: (d) £1.00.
add 15% VAT to the total order value. For fast delivery telephone your order quoting VISA or Access card or official order number.
(Minimum telephone order £5).



• NEWS •

Software famine and hardware glut, counter plots in the QL's advance.

OPD: ICL's Teleputer

December 1981, ICL announced that it was to develop a low cost integrated terminal/digital telephone workstation in conjunction with Sinclair Research. Christened at the time the "One Per Desk IT Work Station" the machine was to incorporate Sinclair's Flat Screen technology.

Thursday the 15th November saw the project's fruition. The flat screen has been dropped in favour of a conventional monitor. Nevertheless, ICL's OPD incorporates a 68008 microprocessor, 128K RAM and twin Microdrives – the hallmarks of a QL. Yet QL it is not.

Simply plug your OPD into a standard British Telecom socket and you have a miniature communications system with two lines for voice/data transmission. Capable of taking advantage of all features of your office's internal PABX phone system, the OPD adds a few of its own.

As an advanced phone it permits short code dialling, automatic answer/redial and instantaneous access to a directory of more than five hundred entries. As an intelligent data terminal the machine allows one to access remote databases, private or public Viewdata, electronic mail and other information services. (ie Telecom Gold, PSS, Prestel, ICL bulletin and mainframes).

The additional hardware to take care of all aspects of telephony includes a 3rd custom ULA, BT approved handset, integrated AMD 7910 chip modem providing data communication on either line conforming to CCITT and BELL standards and even a speech synthesiser with a 152 word vocabulary and the delivery of an articulate Dalek.

Not surprisingly, as regards firmware ICL have done away with QDOS and installed their own system. Fully multi-tasking and with an extremely friendly menu driven user-interface, the system occupies some 144K ROM.

In terms of conventional expansion the OPD offers little in comparison with the QL. A single RS423 port is built-in for use with a printer and two ROM sockets house any additional software. No provisions appear to have been made for networking, RAM extensions or Disk drives. Indeed, all communication with external devices would appear to rely exclusively on the modem.

Aside from a toned down version of SuperBasic known as OPD BASIC the machine comes without any freebie software. However, Psion's Xchange suite is available on a Plug-in ROM module at a cost of £150. All four applications are fully integrated and instantly accessible.

Furthermore, three dedicated function keys – START, RESUME and REVIEW provide the user with an easy way to switch in and out of a number of tasks or applications running concurrently. It's understood that Psion will be developing more ROM based software for the OPD. This will include a improved messaging system enabling information to be transferred from remote databases directly from screen to their four packages.

With emphasis upon task hopping and communications the OPD is aimed at the busy corporate manager whose day is spent talking on the phone, writing memos and searching for information. A market which ICL feel amounts to some 600,000 units and has yet to be tapped.

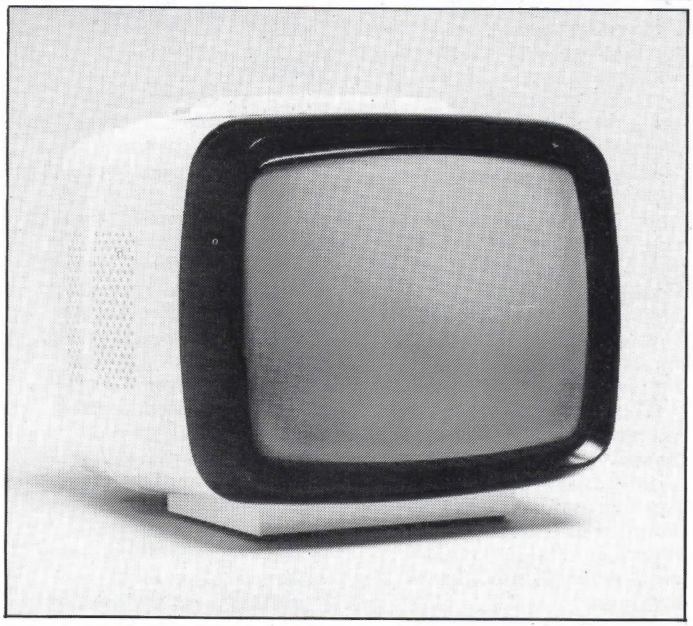
Certainly, with a price tag of £1200 the OPD has no direct competition. However give an enterprising independent a QL and ...

Artistic Talent

Talent Computer Systems already have two quality adventure games (*Zkul* and *West*) on the market. Quite an achievement bearing in mind the logistics of selling on microdrive.

Not prepared to sit on their laurels, Talent are currently developing a third offering. This will be a graphics package similar to Panorama (H) for the Commodore 64. Taking into account the higher resolutions obtainable on QL this could well prove to be spectacular with advanced features such as magnification, free hand sketching and texture definition. Whatever the case, it is bound to start an avalanche of graphics adventures for the QL.

Talent, however, will not have a free rein. CP Software who demonstrated their Bridge simulation at the ZX Microfair are also understood to be developing a graphics package of their own. We eagerly await the results.



The ergonomically-shaped Citadel monitor.

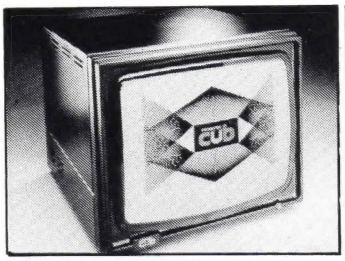
Monitoring A Review

Following on from our monitor review in the October/November issue, Citadel have just announced that their model 101QL monitor is now available.

The reason for the delay is fairly patriotic – the monitor is manufactured in Edgeware and uses over 95% of its components from European sources. The company have had some trouble in tracing low cost components to compete with imports from the Far East. The monitor will cost £69.50 plus VAT, with carriage an extra £5.75. A monitor lead is also available at £5.95 plus VAT. And for those who prefer contrast to compatibility – the cabinet is in black and white.

Apparently the monitor has been around for the last three years, with the distinction or otherwise of being supplied to the MOD and BT.

Citadel Products Ltd, 50 High Street, Edgeware, Middx HA8 7EP.



Cub out in the cold?

Cub Cult

Microvitec's Cub Monitors are to be put through one of their most severe tests ever – encountering climates with excessive cold, humidity and tropical heatwaves.

Microvitec are taking place in a four-year expedition on six continents in which 4,000 'Venturers' aged between 17 and 24 will take part. The scheme features scientific, conservation and community projects on land and sea, with some of the Venturers engaged in archeological and medical research activities. Three of the units are on board the expedition's flagship, "The Sir Walter Raleigh" while a further four will be used in projects, the locations ranging from bush and jungle to desert and ice-cap. Still more of the units will be employed in Operation Raleigh's London Headquarters, in the Hull Communication Centre and by scientists at Leeds University.

Microvitec are currently sponsoring The Cub National Schools Computer Challenge to encourage computer literacy and have provided a monitor. Data from Operation Raleigh will be received by satellite for analysis.

Microvitec PLC, Futures Way, Bolling Road, Bradford, BD4 7 TU (0274 390011).

Reliability OR Credibility

Ablex Audio Video, one of the country's biggest custom duplicators are currently copying programs onto almost 100,000 microdrive cartridges a week. For this they are using 50-60 special QLs. Cartridges are formatted 6 times, auto-recorded, verified, soft-sector tested and so on. All of which ensures, so we are told, 100% duplication quality.

Bearing in mind problems encountered in cloning some of Psion's programs, something however must be at fault. And if not the medium, then the mechanism.

Certainly, on that score, at the OPD launch ICL mentioned that they had improved the QL microdrives to give 99% reliability. Confirming this, Nigel Searle added that most of the improvements had been fed back to Sinclair Research with the result that the drives on the QL were "just as good as the ones on the OPD". And, as if to indicate their faith in the device, Sinclair's MD went on to mention that they were looking into developing a "1 Megabyte Microdrive".

Finger Poppin' Good

Every aadventurous QL owner is a secret key-popper. The Enter key is the best place to start; once that's out, the rest of the keys can be popped with no more than an inquisitive finger nail.

Apart from the surrealist effects obtained by spelling out messages across the keyboard, key popping also gives an insight into that unique characteristic of QL typing – the disconcerting change in required dab pressure between the centre and periphery of its keys.

Needless to say, the bigger the key, the bigger this pressure gradient between centre and rim, until eventually – at the periphery of the two largest keys, Enter and Shift – the required downward force can exceed the powers of even the stoutest digit.

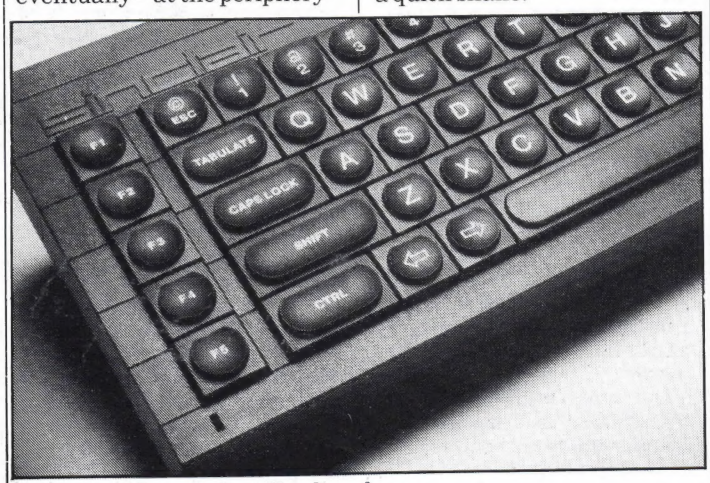
Now, as any QL key popper knows, each key sits on a plastic spigot which passes through a plastic tube on its way to the Spectrum-style touch membrane. It's friction between these two plastic surfaces (I think) which is responsible for the sticking of mis-hit QL keys.

One solution for the problem was proposed by a Sinclair man. 'Grease,' he said tersely. 'Works wonders.'

He may be right. I've even tried Sturmey Archer cycle oil on my Enter key , and found a great improvement – till the liquid trickles away somewhere into the depths of the computer.

But I've always thought that Sinclair could easily and permanently solve this problem by using a spigot made of 'slippier' plastic, and was pleasantly surprised to find that the QL-style keyboard on the new Spectrum + uses this precise method to produce an infinitely smoother typing action.

However, after playing with the £180 machine I've discovered why I'd never make a production engineer. The key tops of these QL-type keyboards are, after all, only a push fit on their spigots, and the same slippiness which makes the spigots move more readily in their tubes also ensures that, after five minutes use, the Spectrum + can be turned upside down and divested of its Enter and Shift keys with no more than a quick shake.



Sinclair QL and the offending keys.

Powerful Gem

Latest gem from Technology Research is a QL Printer Interface. As the QL is only fitted with a Serial for the printer facility this obviously limits choice. So Technology Research have come up with a printer interface for the Centronics type. One cable fits into the Serial Port 1 of the QL and then has a Centronics type plug for the printer. The interface obtains its power from the QL and therefore requires no external supply.

Lightweight, and measuring only 70 × 50 × 25 cm, the printer interface has an overall cable length of about 3 metres and costs £49.95. Once attached, the Baud rate is set to 9600 Baud and is compatible to the default Baud rate of the QL. Print out from Easel, Abacus etc will them work without requiring any changes.

Technology Research Limited, Unit 18, Central Trading Estate, Staines, Middx (Tel. 0784 63547).

Plugging CST

To help promote and maintain QL's sleek and glossy image, Cambridge Systems Technology, specialist peripherals manufacturers, have produced a range of black plastic boxes for hardware and software houses to package their products in.

The company had invested in manufacturing the casings for their own peripherals, such as the Q-Pi Centronics printer driver and the Q-488 IEEE-488 interface. But there would appear to be an excess, so CST have decided to take advantage of the "Black is Back" phase and produce them for the benefit of other peripheral designers working on QL projects. Quite a godsend for those unable to afford alternatives.

The boxes are available unmarked, matching exactly the style and colour of the QL. They can be plugged, via a suitable connector, into the expansion socket on the microcomputer or into the Sinclair expansion module.

Cambridge Systems Technology, 30 Regent Street, Cambridge (0223 323303).

STOP That File

Digitex Computers Limited have just launched a useful software package for the QL. Called STOP (standing for STorage OPtimiser) it will enable microdrive files to be compressed to about half their size.

The aim is the prctical and thrifty one of giving users a cheaper way of storing information on the relatively expensive microdrives.

In addition an upgraded version will be released for use on Winchester systems, which will enable complete back-ups to be made onto microdrives of floppies. *Fini le probleme* of how to back up a hard disk on the QL, and it costs £17.95.

Digitex Computers Limited, 4 Amwell House, The Woodlands, Iselworth, Middx.

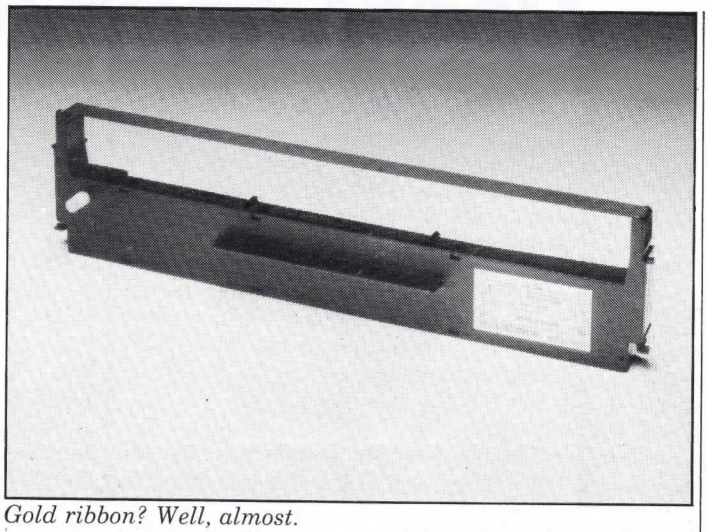
A real investment.

Spike Spoiler

A new item can be added to the list of little extras that make life more bearable. Power International have brought out a device that eliminates spikes in the mains supply.

All you have to do is swap your conventional plug for, wait for it . . . 'The Plug' . . . and then plug-in!

At £17.75, this may appear a little steep but anybody who has had the QL suddenly freeze on them will realise what a devastating effect electrical hiccups can have. So this neat solution is a must for true peace of mind. The two we've got in the office are in constant use.



Gold ribbon? Well, almost.

Long Life Ribbons

For those with more words than ink, Rover Ribbons comes to the rescue.

They have designed a long-life cassette ribbon for Epson's FX80 and RX80 printers which can be obtained from Action Computer Supplies. Stronger and longer-lasting than the standard, this ribbon development can carry enough ink to print over twice the number of characters normally available from a cassette.

The secret of this ribbon lies in a recently-developed weld technique. This not only makes the ribbon stronger but lowers the profile of the join of the ribbon, thus reducing wear to the matrix head.

The unit price is £3.80 for three to five cassettes, with the price dropping, the more you buy. So for aspiring novelists and database fiends, if you bought 24 cassettes or more, it would cost £2.80!!!

Action Computer Supplies, 24 Windmill Road, Brentford, Middx.

Computer Hot Line

British Telecom, Bradford, are claiming to be providing the country's first computer information service (What about Micronet!!) Callers dialling Bradford 722622, will hear a 3 minute tape giving all aspects of computer ownership. Hard to believe, so we gave it a go!

The message gives details of a few new products and a project to compile a 1986 Doomsday Book. (Nothing about the QL!) It ends with a number the caller may dial for further information (0422 842525). Dialling that we were told that sending a 50p postal order to "Information Unlimited, Freepost, Hebder Bridge" our questions would be answered.

The hot line would appear to have gone cold!

Economies Of Scale

At the OPD launch it was said that microdrives offered inherent advantages, ie they were "less expensive, smaller and lighter" than their alternative. However, it was not pointed out that byte for byte the cartridges are 3-4 times more expensive than floppy disks.

When asked, at the OPD launch, whether increased demand might result in the cost of the microdrives being reduced to a sensible level, Nigel Searle, Sinclair's MD simply observed that to do so would only increase demand still further, a basic lesson in economics maybe! Certainly a subtle reminder that Sinclair as the sole suppliers of the cartridges call the shots. But with disk drives and, perhaps, a cheap cassette interface on the horizon . . . who knows?

All I Want For Christmas . . .

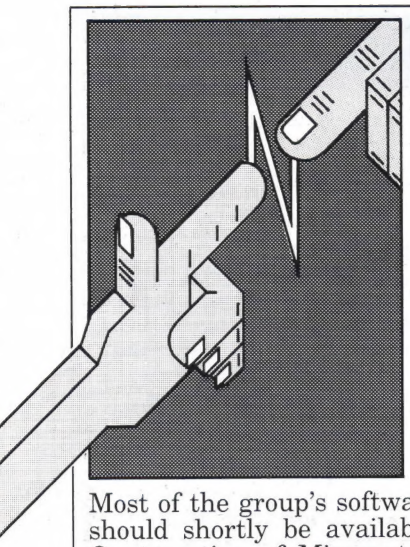
The two lucky people on the right recently won a QL in a competition run by Streetwise Micro, in conjunction with the AD Lib column of the Standard.

But Ruth Pyatt looks slightly more pleased than does Neil Froggatt. Perhaps he really had his eye on Topo, the world's first domestic robot!

The competition was run over three days in the column with a different question each day. First prize along with Topo was an Apple II with which to program him, together worth some £2,500.

Streetwise Micros, London's latest micro computer store, is located in Lion House, Tottenham Court Road. A special prize-giving ceremony was held there hosted by Capital Radio DJ Chris Tarrant.

Lucky winners and their prizes.



USER GROUP

With his finger on the pulse IQLUG Chairman Leon Heller reports on the group's activities and uncovers some murky secrets.

User Group News

Most of the group's software library should shortly be available on the Qnet section of Micronet 800. The software will be generally available to all Micronet subscribers, you don't have to belong to IQLUG but if you do you will receive an £8 voucher towards your Micronet subscription. Existing IQLUG members who subsequently join Micronet will get their first quarter's subscription free. The library will also be available on the Association of Computer Clubs' Clubspot database.

Maverick Member

We were a little concerned about one of our IGLUG members who is advertising QL utility programs for sale that seemed supiciously similar to some of the programs in our library. Another of our members bought the programs to check them out. He reported back that the programs were so badly written, that they could not have had anything at all to do with our library software!

Still on the topic of poor software, one supplier of an alleged Quill "mail-merge" program, that one of our members bought, is using the purchasers to find the bugs! The member in question, finding the program didn't work (it was written in Super-Basic, by the way) telephoned the supplier to complain about it, and heard to his amazement that the supplier had already been notified of 30 bugs! Apparently, when all the bugs have been found, they will be eradicated, and everyone's copy will be upgraded! Of course, there is nothing new about this programming technique, both Sinclair and Psion seem to have adopted it!

Weekend Work-Out

The IQLUG Swindon workshop went very well, with around 100 members turning up on the Saturday at the Wiltshire Hotel, to hear presentations by such luminaries as Tony Tebby, QDOS designer, Dick de Grandis-Harrison of Psion, he wrote most of the documentation for the Psion software, and Chris Scheybeler, of GST. Prototype disk drives (they were actually working!) for the QL were demonstrated by GST. They will probably be launched officially in December. Yours truly gave a marathon five hour session on assembler and QDOS for beginners, at the end of which both the speaker and the audience were throughly exhausted. Some members got so engrossed with what they were doing that they didn't get to bed until 5.30 in the morning!

QL Compatibility

Affluent users who also possess an IBM PC or one of its clones, with the Psion Xchange program suite, will be pleased to hear that programs written on the QL in the Archive programming language may be used on the PC, with very few, if any, changes required. I am indebted to Chas. Dillon for this information.

Recycled Hardware

In addition to the QL, I also possess an antedeluvian 48K Model I TRS-80, equipped with a printer which only has a parallel interface. Being of a somewhat stingy disposition, and not wishing to fork out for a serial to parallel conversion unit, I use the Tandy as a serial to parallel converter. When time permits I will write some software for the Tandy that will turn it into a cheap, if bulky, 48K print buffer!

In Confidence

Several fortunate users have been given beta-test (not necessarily "better" although one hopes so!) copies of the Psion programs. They are vastly improved, by all accounts, especially Quill, which no longer crashes with such disturbing regularity, but is still as slow as ever when doing block moves, I am told. Psion still seem committed to their policy of providing users with hours of harmless amusement, finding bugs in their software, as just after the testers received the new versions, they had a telephone call from Psion telling them that it was full of lots of new bugs! I asked a Sinclair spokesperson about the upgrades, which appear to be common knowledge. She responded angrily with the words "gross breach of confidentiality!" and asked me to reveal my sources of information, which I naturally refused to do. I gathered that the aforementioned expostulation did not refer to me, but to Psion, for giving out copies of the software without insisting that the testers signed confidentiality agreements. Apparently everyone concerned has now been told to deny all knowledge of any upgrades . . .

Delivery Dates

Sinclair still seems to be up to its old tricks. In the second issue of QLUB News, QLUB members are told that the Psion chess program is now available from Camberley. On telephoning Camberley, an IQLUG member was first told that they didn't know anything about it, but on speaking to a supervisor, was told that it would be 28 days before it would be dispatched! I seem to have heard that before somewhere . . .

Postage Refunds

Many users who have returned their machines to Sinclair for repairs or upgrades are not claiming a refund of the postage costs from Sinclair, which they are of course entitled to. Since the record (unless someone knows better) for the number of times a machine has been returned stands at five, some large sums of money might be involved for many users. Sinclair seems to pay all claims in full, without any quibbling.

Another thing that Sinclair seems to forget to tell people about is that the latest manuals (with 6/84 on most of the sheets) are available free of charge from Camberley. Just give them a ring, and they will put a copy in the post to you.

IQLUG is a non-profit making group, with officers democratically elected by members at an Annual General Meeting. Accounts are independently audited and made available to members. The group is affiliated to the Association of Computer Clubs.

Membership is by subscription to Quanta, the group's monthly newsletter, currently containing 28 pages of members' letters, hints and tips, news on the QL scene, program listings and so on.

The group maintains a (mostly) free software library, which currently contains 15 programs. All library software is written by our members, and only non-commercial items will be held there. In addition a free advice service is provided: members can phone in with their problems, and be put in touch with someone who can help them – a register of members with expertise in various areas is kept.

Membership details are available from: Brian Pain, 24 Oxford Street, Stony Stratford, Milton Keynes MK11 1JU.

QL

£39.95 INC

PARALLEL PRINTER INTERFACE

- ★ 12 months guarantee
- ★ Fully self-contained with connectors and 1·5 metre cable
- ★ Plugs into Sinclair QL's RS232C port and
- ★ Drives any CENTRONICS compatible printer, eg Epson, Seikosha, Juki, OKI, NEC, Shinwa, Star, MCP-40, Canon, etc, etc
- ★ Fully QDOS and SuperBasic compatible

To order send name and address with cheque to:
MIRACLE SYSTEMS Ltd
Avondale Workshops
Woodland Way
Kingswood
Bristol BS15 1QL

Sinclair and QL are trademarks of Sinclair Research

0272 603871 Ext 210

Available at larger branches of WHSMITH
Access orders welcomed

QL COMMPAK

MULTI STANDARD ACOUSTIC MODEM AND PRINTER CONTROLLER ONLY £140 INC.

Plugs into SER2 port of QL using QL Serial Cable (not supplied). Accepts phone handset for quick and simple connection.

Modem Speeds — 300/300, 1200/75 (PRESTEL), 1200/1200
Standards — Bell 103/113, CCITT V21, CCITT V23
300 Baud Modes — Originate or Answer
1200 Baud Modes — Standard or Equalized (for noisy lines)

On board Microcomuter controls speed conversion, buffering and Modem operation. EPROM firmware allows for future developments. In 1200/1200 Mode, Micro uses 'Packet' techniques to avoid collisions and allows fast data transfer between COMMPAK Users.

Printer mode provides Serial to Centronics conversion. uses standard printer cable (available at £15 inc.). Modem 'Log' function copies 300 Baud line data to printer.

Comes complete with instructions and QL Terminal Emulation software package on microdrive. Please send for details.

Make cheques/POs payable to COMMPAK DATA. Prices include VAT and p&p. Allow 28 days for delivery.

COMMPAK DATA
13 Beechwood Road, Uplands, Swansea

12 YORK PLACE
BRIGHTON
SUSSEX
BN1 AGU

MICROWORLD
COMPUTER & VIDEO CENTRE

10 THE BOULEVARD
CRAWLEY
SUSSEX
RH10 1XX

Best prices, best advice and support

TOP PRINTER NAMES FOR YOUR QL

TAXAN KAGA — PRICES INC. VAT

KP810 (150 cps and 29 cps NLQ) £295
KP910 (As above but 132 Col.) £399

EPSON

RX 80 F/T + (100 cps) £249
RX 100 F/T £398

SMITH CORONA

Superb range of rugged printers with dual parallel/ Centronic interfaces, Epson compatible, 2K buffer, 160 cps draft & 40 cps NLQ printing.

D200 (80 Col.) £334
D300 (132 Col.) £559
D100 (120 cps, 80 Col.) £239

SHINWA

CPA 80-Q (100 cps for the QL) superb printer £279

MICRO PERIPHERAL

MP 165 QL (165 cps draft, 75 cps NLQ) for the QL computer £419

DAISY WHEELS

Quendata (20 cps) £279
Juki 6100 ..... £389
Juki 6300 (40 cps, 3K buffer) £880

Note: Serial to parallel interface only £38

TOP MONITOR RANGE

SINCLAIR VISION QL

BEST PERFORMANCE

- •RGB
- •>18 MHz bandwidth
- •640×440 dot resolution
- •12in non-glare screen
- •Superb engineering £229

MICROVITEC QL

BEST PICTURE

CUB 1451/DQ3
- •14in, RGB
- •Top quality
- •Versatile & matching £275

PHILIPS

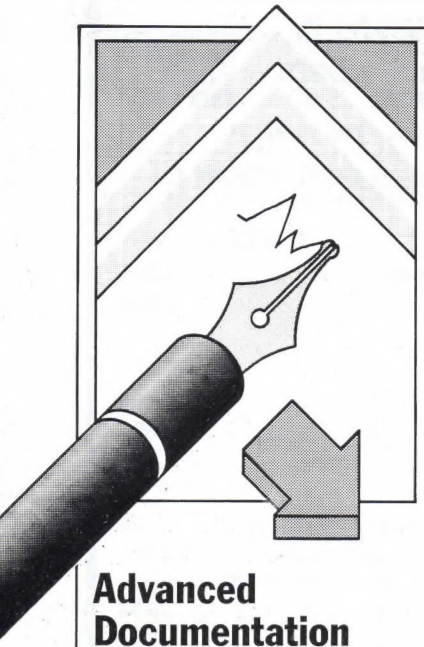
- •12in
- •2000 character display
- •Green Phosphor
- •Super price £89

QL Computer £399

Rang 5/3/85.

PLEASE ADD £7 P&P, CARRIAGE TO YOUR ORDERS

Telephone orders:
Crawley (0293) 545630, (0273) 671863 Brighton
(0273) 69841 or Farnham (0252) 726379
VISA & ACCESS welcome
Export orders welcome



OPEN CHANNEL

This is the spot where we turn the magazine over to you, our readers. We welcome any comments, criticisms or anecdotes about either the QL or QL User.

The address to send your letters is:

Open Channel, QL User, Scriptor Court, 155 Farringdon Road, EC1R 3AD.

Advanced Documentation

Adam Denning's article (Oct/Nov) mentions QDOS documentation. Where can this be obtained? A full list of addresses for vectors and subroutines would make a useful and interesting feature for this magazine. What use is an assembler without subroutines for the keyboard input and screen output?

Neil Street
Greenock

Full documentation is available from Tony Tebby, QDOS's author at a cost of £35 inclusive. Write to: QJUMP, 24 King Street, Rampton, Cambridge CB4 4QD.

In addition, we recommend the 'QL Advanced User Guide' by Brian Dickens, reviewed in this issue.

Finally, 'Advanced QL Machine Code' written by Denning himself and published by Duckworth's (£12.95) provides detailed lists of addresses and vectors as well as a number of extremely useful routines. Well written and comprehensive, the book represents the definitive guide to QDOS and is a must for the machine coder.

In The Light of Experience

I would like to bring your attention to some interesting points. First I must say that I believe the QL to be a fine computer, although the after sales service leaves a lot to be desired, especially for mugs like me who joined QLUB.

I recently discovered something that does not seem to be written up in the manual. When using LIST the SCROLLing action can be PAUSEd by pressing CTRL & F5. To continue the LISTing just touch any key except CTRL, this can of course be repeated with one LIST.

There have been many suggestions of early models being updated free with modified QDOS. I believe under the law of contract any goods which are sold faulty may be returned within a reasonable period (in excess of 12 months). I have just been informed by a Sinclair technician, when inquring about a fault in my PAN command, that it does not work properly in any of the released versions.

Now to QLUB – if you want software updates wait until they are available so the maximum benefits of an annual subscription can be gained. I have been a member for 4 months – I have not received a membership card or any newsletter, although given assurance 5 times on the phone that a memo would be passed on by Customer Liason.

As for solving customer problems with software, I was given several possible solutions to each particular problem. These solutions were standard retorts in text book language for which I had to ring Sinclair to decipher. Whilst I do not want to be treated like a child, I do not expect to be treated like a compulsive text book reader.

If some are concerned that by not joining QLUB they will not be able to correct any faults in software, Psion should be liable for selling faulty goods.

I must mention that Psion have kept their promise to return enquiries within 48 hours. Sinclair took 16 days (promised 7) to return Easel, and that was after shouting down the phone to a cocky bloke in the Liason Department, who laughed when I told him I'd been waiting 14 days.

Talking of promptness your magazine came only 14 days after the stated date of arrival.

A. L. Towers
Nether Edge, Sheffield

Without going into too much detail, there are three 'implied' contractual terms spelled out in the 1979 Sale of Goods Act. These are in effect promises made by the trader to you about the quality of goods you are buying. They are: that the goods are fit for their 'usual use'; that the goods are 'as described', and that the goods are of a proper 'merchantable quality' – so if you buy something 'new' it should be in perfect condition.

These three terms should cover the most frequent problems consumers face, ie 'it doesn't work properly', 'though there's nothing wrong with it, it's not what I asked for', and 'it's of poor quality'.

As long as these facts fit your case, the law is on your side.

Double Standard

One of the criticisms that has been levelled at Quill is that you cannot save a standard format for a document ready to use straight away, but have to spend time resetting the margins, the footer and the display at the start of the new document.

But this is not so. If, after you have done your setting up, you save it under say: **fmt_doc,** and remember to change the name of the file when you save your text, the format will be there ready for the start of next document. This works because Quill files do not need any text content.

One could also have a standard format ready with your address at the top right already – or anything you like.

Roger MacNicol
Oxford

Standard formats are useful but only for 'one-off' tasks. The real criticism levied against Quill has been its failure to allow for the production of 'standard letters' or 'forms' for mail shots etc.

Drive Deride

I have a JM version and on the whole it seems to work. Your correspondent, Capt Charles, is correct when he says the merge command locks up the computer. I had that very problem when I tried to merge a short document into the end of a long one by loading the long document first. I was also unsuccessful when I tried to merge the short document in from mdv1. I finally succeeded by loading the short document, positioning the cursor at its beginning and then merging the long document into it. After about 90 seconds of frantic spinning on the part of mdv2 the screen dissolved into a correct combination of the two documents. I don't know why it worked that way round.

I have also had trouble with the save command when I find it hasn't saved the last part of a document. I wonder if it relates to the micro-cartridge being over half full, and one is working on a fairly large (1500 words) document; or if it is just that I am wearing red socks that day, because at other times the machine works perfectly.

The QL deserves to succeed because it is the first cheap computer that actually does something useful (the Beeb is very good but one needs to

spend much more). The cartridges are still overpriced in the shops (at least one can get them now) and that really limits the machine's usefulness. Byte-for-byte they must be far-and-away the most expensive form of storage. I cannot agree with your contributor that microdrives should be an intermediate measure until the price of disk drives fall. Microdrives are the main reason for the QL's existence. Substitute disks and the machine would be twice the price and then one might as well go out and buy something with a proper keyboard.

Finally, Sinclair should be encouraged to release the updated software to existing users as soon as possible, I think it is in their best interests.

Dr John Heckmatt
Ealing, London

Failsafe

The following steps not only should stop Quill saying 'Failure while writing to edit file' when trying to use the commands, but can also shorten the accessing times for commands (eg the 'Design' command takes 5 seconds instead of 8 for accessing): –

1. – After using Quill put the backup cartridge in mdv1_ having taken the Quill cartridge out (get into BASIC first by pressing reset).
2. – Type in: LOAD MDV1_def_doc (This file will already be on it).
3. – Put the backup cartridge in mdv2_ and Quill in mdv1_ and then type: MRUN mdv1_boot
4. – Repeat this every time while loading Quill.

Quill will now load and it should be faster than before in accessing commands.

M. Satchi

A Positive Approach

I have just received Issue No. 2 of the QLUB Newsletter, the new format is a great improvement on the old, and the content is also more useful and informative.

I received a copy of the Metacomco Assembler Development Kit about a week ago and have found it to be an excellent product all round. The first routine which I decided to write was a screen to printer copy instruction that would be linked in as a BASIC keyword. After spending the best part of a week getting the routine into a slightly working form, having no information on QDOS this was rather fun, I had the nasty dawning of a suspicion that the screen copy routine from EASEL might be extractable. This idea came about upon reading the article on page 5 of the above mentioned newsletter.

After a bit of fiddling about I came up with the following short BASIC program which, with it's associated loader, will allow this screen copy routine to be added to the BASIC interpreter as a new keyword. A quick explanation of the routine follows: –

Firstly, space is reserved to contain the original Easel file plus enough for the extra code required. Then this code is read into the reserved area of memory with a standard checksum loader (this one was stolen from the clock routine in the back of the newsletter), the original Easel file is then copied in at the appropriate point, then the new routine is copied onto a new file called 'SCOPY_PRC'. Obviously the microdrive device numbers can be swapped about as required.

To get the thing up and running it only requires typing in the BASIC code and then placing a formatted microdrive cartridge in drive 1, and the Easel cartridge in drive 2. Typing 'RUN' starts the whole thing off, if you have typed any of the data in wrong: – you will be so informed, if not, then the drives will whirr, the lights will flash, and before you know where you are your Screen Copy routine is resident on the cartridge in microdrive slot 1!!!. Below is the BASIC listing, which has been imported into QUILL and should, therefore, be fault free.

I have found it necessary to send some control codes to my printer (ESC, '1' on my Epson RX80-FT) to get this routine working with Easel and as a stand alone. These codes set the printer to 7/72nd inch linefeeds and avoids gaps between each pass of the printhead. I have not modified the linker routine to include the printing of these codes as they may vary from printer to printer, though this should be a simple matter for anybody with the right knowledge of QDOS and machine code. One slight problem with this routine is that it seems to miss out the bottom few pixels of the display. This may be on my version only or not, but I am working on it to find out if it is possible to modify the routine to get rid of this minor mishap.

```
100 a = RESPR (400)
110 RESTORE
120 FOR byte = 0 TO 39
    STEP 8
130   cs = 0
140   FOR o = 0 TO 7
150     READ x:cs = cs + x
160     POKE a + byte +
        o,x
170   END FOR o
180   READ x:IF x<>cs
      THEN PRINT
      "Error in
      line"!byte/8*10 +
      1000:STOP
190 END FOR byte
200 LBYTES
    mdv2_gprint_prt,a +
    34
210 SBYTES
    mdv1_scopy_prc,
    a,350
1000 DATA 67,250,0,12,52,
     120,1,16,518
1010 DATA 78,146,112,0,
     78,117,0,1,532
1020 DATA 0,18,5,83,67,79,
     80,89,421
1030 DATA 0,0,0,0,0,0,0,0,0
1040 DATA 0,0,0,0,0,0,0,0,0
```

For those amongst us with different versions of EASEL, the 'gprint' in program line 210 should be replaced with 'sprint' or whatever else is appropriate.

To get your new routine into the machine it is neccesary to use a routine like the one below: –

```
10 LET a = RESPR(350)
20 LBYTES mdv1_scopy_
prc,a
30 CALL a
```

Michael Benn
Tunstall, Sunderland

Brother In Arms

I purchased a Brother EP 44 printer in anticipation of the receipt of my Sinclair QL which arrived towards the end of August after a 6 month wait. I was however delighted to receive the QL but most disappointed when I could not get it to print properly on the EP 44.

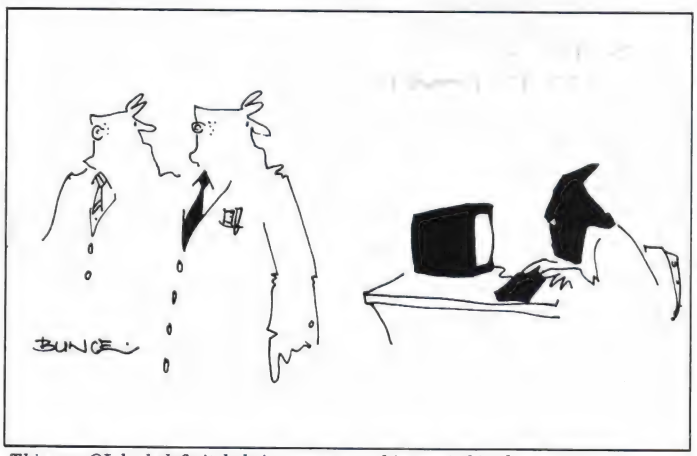
Being a member of QLUB I wrote to ask for a printer driver for my EP 44. I got a prompt answer on how to link a Brother HR-5 which was of little use. Also the information given was not very comprehensive or given with much confidence – 'from discussion with Brother engineers we believe that the correct pin combination should be'.

Other than this uncertain and as it happens useless information I have received nothing for my QLUB subscription in three months. Can any readers help with a detailed printer driver with all parameters and an explanation how to print the £ sign.

What about more guidance in *QL User* on how to set up complete systems and how and where one can check before buying peripherals to ensure that they are all compatible. I bought my QL by mail order direct from Sinclair and therefore had no opportunity to see a complete system working before my purchase. I suspect a lot of readers will have had the same kind of problems.

J. R. G. Stephenson
Jeddah, Saudi Arabia

We are currently compiling a detailed list of dip switch settings and driver control codes for the more popular printers. Any help from our readers would be greatly appreciated.



This new QL look definitely brings out something weird in the operators.

SATELLITE STATIONS

PHOTOGRAPH BY GEORGE LOCAN

Adam Denning reveals the secrets of QL communications.

A standard requirement of modern computers is the inclusion of various standards of communications protocols. The most common of these is the RS232-C standard, which uses a Canon 25-way connector at either end and sends data between the machines serially. This means that rather than handling each datum as a number directly, it is split into its binary component digits which are sent one at a time. These data bits are 'framed' by a start bit and a stop bit, to tell the receiving device that an item of data has been sent. Naturally, the sender and receiver have to be using the same transmission speed, otherwise the data will be garbled and meaningless. To this end, various standard rates are defined, usually ▶

called 'Baud rates' (this is approximately equivalent to 'bits per second' and refers to the number of binary digits which can be sent along the line in a given space of time).

Even though two communicating machines will be talking to each other at the same rate, often the receiving machine cannot cope with the data at the rate it's being received at. It may be executing some complex processing on it, for example. When this happens, the receiving machine has to be able to tell the sending machine that this has happened so that the sender can stop sending data for a while. This is done by a process known as handshaking, which normally involves two wires, called RTS (for Ready To Send) and CTS (Clear To Send). When the receiver is temporarily incapable of reception, it sets its CTS line high. This level is transmitted to the sender (to its RTS line), which interprets it as meaning 'Stop!'. When the level on CTS goes back to its normal value, transmission starts again from where it left off.

There are problems, of course. First of all, everyone seems very capable of interpreting the definition of the RS232 standard, but everyone also seems just as capable of interpreting it in a different way. Lines can get confused, left out and even replaced by others, and voltage levels, although theoretically 12V, can be almost anything from 3V to 15V. A good RS232 interface will be capable of interpreting all this, but to make matters worse some manufacturers have adopted a slightly different standard. Acorn, for example, uses the RS423 standard, with 5V levels, on its serial port. The QL, as it is reasonably close to true RS232, can successfully interpret BBC Micro RS423 data.

To connect a BBC Micro to a QL via the RS232/RS423 connectors, all we need is a Sinclair Research QL RS232 lead (or similar), a 25-way Canon socket, a BBC RS423 DIN plug and some wire. The QL lead plugs into SER2 and the connections are as follows:

25 way PIN 2 (TxD) to BBC PIN 2 (Data out)

25 way PIN 3 (RxD) to BBC PIN 1 (Data in)

25 way PIN 5 (CTS) to BBC PIN 4 (RTS)

25 way PIN 7 (GND) to BBC PIN 5 (GND)

25 way PIN 20 (DTR) to BBC PIN 3 (CTS)

The QL's DTR is near enough equivalent to RTS, and certainly these connections produce the desired effect. With a cable wired up in this way it is possible to use the BBC as more reliable storage medium for the QL, or as some sort of cassette/disk drive interface, or even a printer spooler. It is also possible to have the QL acting as a terminal for the BBC Micro. If you wanted to do this, a few commands need to be issued to the BBC, and the QL would need a terminal emulator program.

Type these commands into the BBC Micro:

★FX7,7 (set transmit Baud rate to 9600)

★FX8,7 (set receive Baud rate to 9600)

★FX156,146,0 (set 8 data bits, 2 stop bits)

★FX181 (enable RS423 key events)

★FX15,0 (clear buffers)

★FX3,1 (send screen output to RS423 as well)

★FX2,1 (get input from RS423)

The ★FX156 command programs the 6850 ACIA inside the BBC Micro telling it to send each byte of data as 1 start bit, 8 data bits and two stop bits. This ensures that the QL receives it correctly. The ★FX181 makes the BBC Micro treat RS423 input EXACTLY as if it were coming from the keyboard; ESCAPEs are noted and function keys are expanded into their requisite strings. The QL program is almost as simple:

```
100 MODE 0
110 BAUD 9600
120 OPEN#3,ser2c
130 REPeat loop
140 REPeat loop1
150 a$ = inkey$: if a$<>'': exit loop1
160 a$ = inkey$(#3): if a$ = '': next loop1
170 print a$;
180 END REPeat loop1
190 REPeat loop2
200 print #3;a$;
210 a$ = inkey$: if a$ = '': exit loop2
220 END REPeat loop2
230 END REPeat loop
```

In this example, neither the QL's keyboard or the RS232 port has priority – the input of one can (and will!) interrupt the input of the other. The Concepts section of the QL User Guide tells how to get various ASCII values from the keyboard, but the useful keys are DELETE (SHIFT-ESC on the QL) and ESCAPE (which, because of the FX181, is ESC on the QL too). All the function keys can be used, but they are fairly hard to get at from the QL keyboard as most of them require the simultaneous pressing of three keys!

There are various manufacturers either planning or producing numerous modems, all of which can be used with the serial ports, but for really effective use we would need a real terminal emulator program. There aren't too many of these about at the moment, but the author has just written one in a mixture of BCPL and machine code and this should be on the market very soon. Modems allow the connection of computers across much larger distances than a single RS232 cable, and there are various information and computing organisations whose livelihoods depend on modems and BT telephone lines. Two of the most apparent are BT Gold and Prestel, although both seem a little prone to security breaches ('hacking') at the moment. The usefulness of either system is debatable, but BT Gold offers the professional user a lot more than Prestel or Micronet can hope to offer as the system gives access to the processing power of the computers holding the BT Gold operating system. At times it makes Prestel seem like a bad joke.

As the RS232 serial ports on the QL can also be used to communicate with other peripherals, such as printers, they are an essential requirement of the hardware. But a current computing craze is networking. This sounds fairly big-business orientated, but in fact any two or more QL owners can do it (and it isn't sinful!). RS232 is a form of networking, but as so many systems interact with their RS232 interfaces in so many different ways that a universal form of 'operation over a distance' or sharing of resources is very difficult to implement. Consequently various manufacturers have come up with what tend to be unique ways of communicating with other pieces of equipment either of their own or of very closely related manufacture.

An early form of this is the renowned 'Cambridge Ring', which is a networking system linking various parts of Cambridge (Sinclair Research, Acorn Computers and the Computing Laboratory of Cambridge University are three reputed users) together, using the British Tripos operating system. Reports differ as to the usefulness of this system, but the general impression is that the people in Cambridge find it essential.

Acorn and Sinclair both adopted networking techniques, Acorn using Econet to connect numerous BBC Micros and Sinclair using another network (with no brand name) for Spectrums with Interface 1s attached. This system was primarily developed so that Sinclair could compete in the educational market where resources are few. The ability to share a printer, and possibly some microdrives, amongst all the members of a class is very beneficial to local authorities' education budgets.

The idea of a network is not just that a computer can transmit and receive data from another computer, as a simple RS232 could be used for

this, but that whole chunks of a machine's operating system can be used to divert file handling and printer serving operations to the network device rather than a local file system/printer. Hence, various machines can share a network and all its resources.

Having had a reasonable success with the Spectrum's network (the main success being that it worked!) Sinclair decided to incorporate the same standard into the QL. Unfortunately it turns out that in practice the connection is unreliable, as although one QL can talk to another quite happily, it seems that slightly differing data speeds cause the QL and Spectrum to ignore each other. There have been reports of QL to Spectrum networking correctly, and indeed the author has found this to be the case, but the obverse is far more representative of the truth, which is probably why Sinclair has dropped any mention of QL to Spectrum networking in its advertising.

It further turns out that the QLAN ('QL local area network) is much slower than 9600 Baud RS232 with no inherent increase in reliability, and as QDOS, being device independent, allows filing system commands to be directed to the RS232 as well as the network, there seems little point in using it. Its one advantage apears to be that, as its protocols are rather more relaxed than RS232's, one station can listen to another specified station, talk to a specified station, listen to anybody and talk to anybody.

QLAN is, like RS232, a serial link, but rather than 'hard-wired' handshaking via RTS and CTS lines, the network is watched over by each machine's operating system. The data to be sent via the network is first buffered, so that each individual transfer is more efficient, and then the buffered blocks are sent out preceded by various numbers referring to the sending station, the receiving station, the number of this 'packet' of data and so on. Reception naturally follows similar lines and considering the simplicity of the system it is disconcerting to find that it is so slow. This cannot be down to the actual transmission rate as it is higher than the highest RS232 Baud rate, so it comes down to hardware and software. Take, for example, the terminal emulator program mentioned earlier. To test it out, I took advantage of my other QL and copied the file to the second QL which was geared up to EXEC_W it. The file is 15000 bytes long and took between thirty and forty five seconds to transfer at 9600 Baud. When it was copied to the network the process took over four times as long.

Another common problem is that despite – or perhaps because of – the QL User Guide, few people seem fully confident in their abilities to use the right commands from SuperBasic. Imagine two QLs connected by a single network lead. Type NET 1 (and press ENTER) into the first QL. We'll call this QL1 (!). Then type NET 2 (and press ENTER) into the second. That bit's easy. Supposing there is a BASIC program resident in QL1's memory, you can send it to QL2 as follows:

Type at QL2:

LOAD neti_1 to load the program, or

LRUN neti_1 to load and run it, or

MERGE neti_1 to merge it with any resident program, or

MRUN neti_1 to merge and run it

Press ENTER on QL2. Now go back to QL1 and type

SAVE neto_2

There's no need for file names.

After a while, the program will be transferred and the requisite action will have been taken at QL2. Now suppose we still want to load one of QL1's programs, but this time it is on microdrive 1, and called test. Type whatever QL2 command you prefer (shown above), and then type

COPY_N mdv1_test TO neto_2

And hey presto! The next case is where we want to send a text file on QL1's microdrive 1 to QL2's microdrive 1. Type the same QL1 command as shown above, and type this into QL2:

COPY_N neti_1 TO mdv1_filename_txt

The technique ought to be becoming clear. To send a screen display from QL1 to QL2, type

SBYTES neto_2,131072,32768

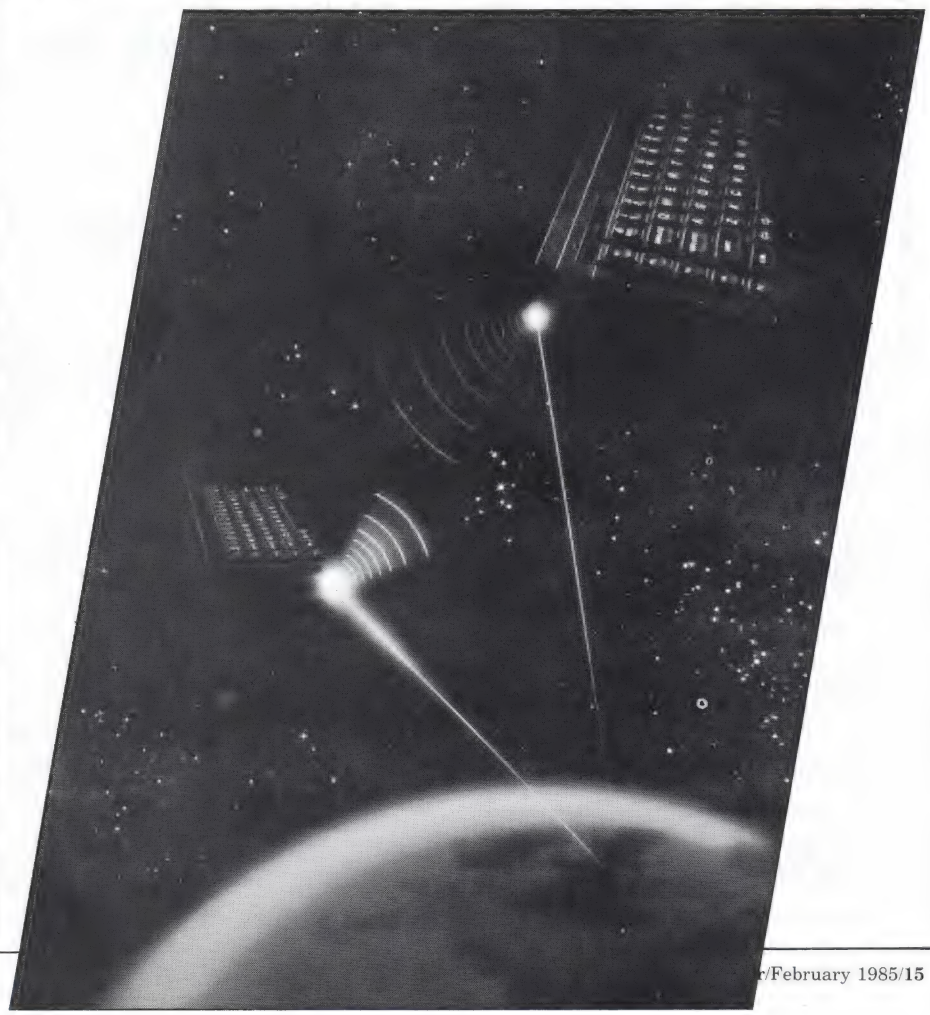
at QL1, and

LBYTES neti_1,131072

at QL2. The same commands apply to all memory to memory code transfers, and code files (to be CALLed) can be transferred using COPY or COPY_N. To send code files which are to be EXECd, we need a slightly different syntax. Normally the file to be sent will be on drive, so COPY mdv1_file TO neto_2 can be used at QL1's end, and EXEC neti_1 or EXEC_W neti_1 at QL2'S end as appropriate. If ever the circumstance arises where you want to send a section of QL1's memory to QL2 as an EXEC file, you need to know a lot about what you are doing. QL1's command must take the form:

SEXEC neto_2, startaddress, codelength, datalength

and QL2's command is still EXEC/EXEC_W neti_1. The command syntax for transmission of all these filetypes via the serial ports is identical except that neto_2 and neti_1 should be replaced with ser1 or ser2. DO NOT use ser1z or ser1c, as both these forms take ASCII 26 (CTRL-Z) to mean end of file, and as this value may well occur in a code file you may find reception ends prematurely.



Machine Code Tutorial

The 32/8 bit microprocessor on the QL is the gateway to real computing. Giles Todd launches this series and delves into the mysteries of machine code programming and the 68008.

PHOTO BY RICK CORDELL

It is recognised that most of the worthwhile software for microcomputers is written in machine code rather than BASIC. So why machine code and what are the advantages of programming in it?

Weighing In

The first is speed. Each line of a BASIC program must be evaluated by the computer in order to decide what the programmer wants it to do, then a series of machine code subroutines must be called to dissect execute the BASIC instruction. The process of evaluating the BASIC line called interpretation usually takes much longer than the time taken for the actual execution of the instruction. But with a machine code program the processor can execute the instructions directly without the need for interpretation.

The second big advantage is that machine code gives you much more control over the computer's resources. Much has been made of the QL's ability to 'multi-task', i.e. the ability to run two or more programs apparently simultaneously. However, multi-tasking programs must be written in machine code – you cannot run multiple SuperBasic programs on a similar 'time-sharing' basis.

Mnemonics

Set against these advantages, are a number of disadvantages. First of all, machine code instructions do not have friendly names like **'PRINT'** or **'LET'** – in fact they are simply binary numbers in the computer's memory. But luckily nobody is expected to program in binary these days. For any significant amount of machine code programming, an assembler is required – this is a program which translates symbolic names (such as **'MOVE'** or **'ADD'** – more on this later) into the binary numbers that the computer understands. However, some useful work can be done without an assembler as we shall see later on.

The second major problem is debugging the machine code. In BASIC you can stop the program at any point and examine the values of the variables to see what is going on. In order to do a similar job with a machine code program, you need a program called a debugger. But, there ▶

are techniques for debugging without a debugger. Assuming you have not been put off already, let's get down to the 'nitty-gritty'.

Brass Tacks

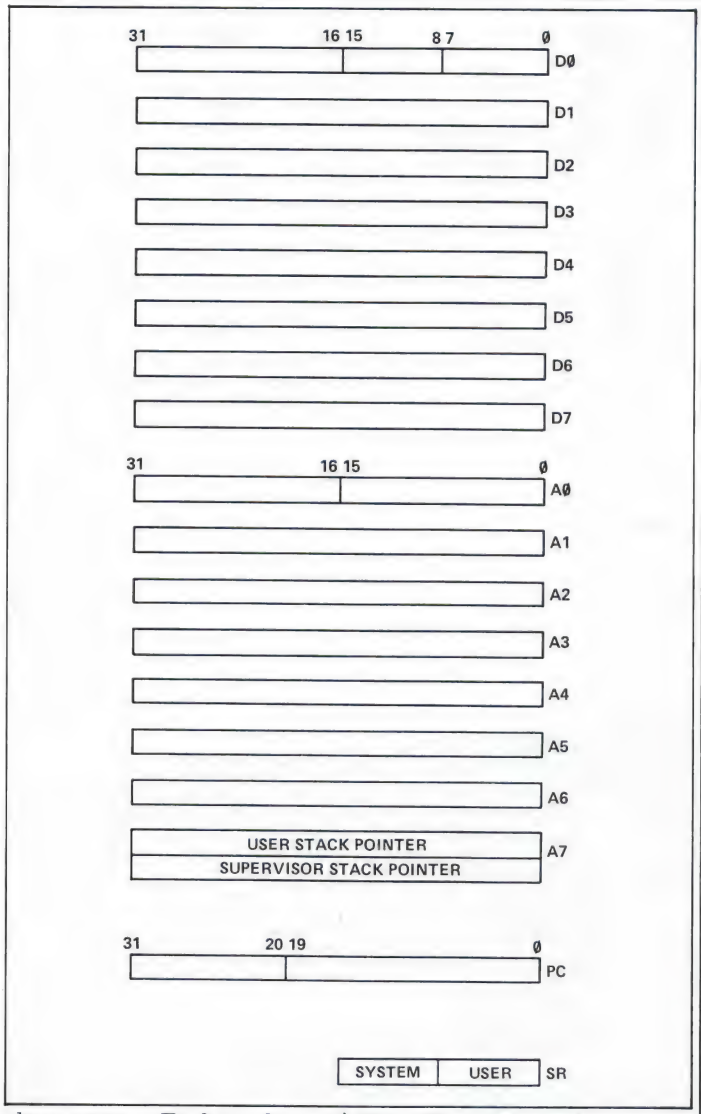
The QL uses a very powerful processor – the 68008. As you can see in figure 1, it offers many more (and larger) programmable registers than any of the 8 or 16 bit microprocessors currently available. These registers are where the computer does all its work. The data registers (D0 to D7 in the diagram) are roughly analogous to 8 32 bit variables on which arithmetic and logical operations may be performed. Note the division of DO into three sections. All the data registers may be used to manipulate four types of data – individual bits, 8 bit bytes, 16 bit words and 32 bit long words.

Whilst some arithmetic operations can be performed on the address registers (A0 to A7), their main function is to facilitate memory access. Many of the 68008's powerful addressing modes require that the address of the stored data be placed in one of the address registers. Note that A7 has a special purpose – it is used to address a particularly important data structure called the 'stack'. The 68008 can operate in two modes (supervisor mode and user mode) and each has its own stack pointer which is referred to as register A7. All the addressing modes and the function of the stack will be explained in this series.

The function of the program counter (PC) is simply to allow the processor to keep track of where it is in memory. Although, like the rest, it is a 32 bit register, the 68008 can 'only' address 1 Megabyte of memory so only bits 0 to 19 of the program counter are used. Similarly, only 20 bits of the address registers are significant when they are being used to access memory.

Hex Loader

Listing 1 is a short program which will allow you to enter machine code programs into the QL's memory as a sequence of hexadecimal (base 16) numbers. Base 16 numbers (which use the digits '0123456789ABCDEF' to signify 0 to 15 and then follow place value rules as in the decimal system) are used in preference to either binary or decimal. This is because an 8 bit number can always be represented as two hexadecimal digits – binary and decimal tend to be more clumsy to use. Each machine code program in this series will be presented in two forms – as symbolic instructions suitable for input to an assembler and also as a series of hexadecimal numbers which can be entered via the program in listing 1.



Advanced Booking

Before running the hex loader program, you should reserve some memory for your machine code. You use SuperBasic's **'respr'** statement for this purpose e.g.:

a = respr (1024)

will reserve 1K of memory. This is **most** important as otherwise SuperBasic will assume that it can use all of the QL's memory – I can confidently predict that there will be unpredictable results!

On running the hex loader, you will be prompted for a start address (in hex). If you have reserved 1K of memory as shown above, then you should enter 3FC00 (subtract 400 hex for each additional 1K of memory reserved). After entering the start address, you will be prompted with consecutive memory addresses and should enter the byte corresponding to that address in the hex listing. If you make a mistake and enter an invalid hex digit, then you will be prompted for the same address again. When you have finished, enter a full stop. The program will then print the start address and the length of the machine code in decimal so that you can save it with the **'sbytes'** statement.

Quartet

This issue we shall take a brief look at four 68008 instructions – **'MOVE', 'ADD', 'LEA'** and **'RTS'.** The example routine is shown in assembler source form in listing 2 (an assembler will appear next month). Actually, all that this 'Noddy' program does is perform the equivalent of the BASIC statement

LET a = b + c

but it will serve to illustrate some important points.

If you reserve 1K of memory with the **'respr'** statement, then the program will start at (hex) address 3FC00 (261120 in decimal) and the data will start at 3FC12 (261138 decimal). Use **'poke_1'** to put two numbers at addresses 261138 and 261142. Now **'call 261120'** and then **'print peek_1 (261146)'.** If all has gone well, the result of adding the two numbers you poked into the computer's memory should be displayed on the screen.

The routine works in the following way. The first instruction loads register D1 with the value contained in

```
* 68008 integer addition
* PROGRAM AREA
    MOVE.L      OPERAND1(PC),D1       * GET THE FIRST OPERAND
    ADD.L       OPERAND2(PC),D1       * PERFORM THE ADDITION

    LEA         RESULT(PC),A1         * POINT TO RESULT
    MOVE.L      D1,(A1)               * AND STORE THE SUM

    MOVEQ       £0,D0                 * NO ERROR MESSAGE
    RTS

* DATA AREA
OPERAND1:       DS.L        1
OPERAND2:       DS.L        1
RESULT:         DS.L        1
```

```
10 REMark Hex loader - G Todd 25 November 1984
20 :
30 CLS
40 hex$="0123456789ABCDEF"
50 REPeat loop
60 INPUT "Start address? ";start$
70 start=hex2dec(start$)
80 IF start<>2^31 THEN EXIT loop
90 END REPeat loop
100 :
110 REMark main loop
120 :
130 REPeat loop
140 PRINT dec2hex$(start,0);" ";
150 INPUT byte$
160 IF byte$="." THEN EXIT loop
170 IF LEN(byte$)=1 THEN byte$="0"&byte$
180 byte=hex2dec(byte$(1 TO 2))
190 IF byte=2^31 THEN NEXT loop
200 POKE start,byte
210 start=start+1
220 END REPeat loop
230 PRINT:PRINT:PRINT "Start = ";hex2dec(start$)
240 PRINT "Length = ";start-hex2dec(start$)
250 STOP
260 :
270 REMark hex to decimal conversion
280 :
290 DEFine FuNction hex2dec(h$)
300 LOCal i,j,decimal,find_hex
310 decimal=0
320 FOR i=1 TO LEN(h$)
330 j=-1
340 IF NOT (h$(i) INSTR hex$) THEN decimal=2^31:EXIT i
350 REPeat find_hex
360 j=j+1
370 IF h$(i)==hex$(j+1) THEN EXIT find_hex
380 END REPeat find_hex
390 decimal=decimal+j*16^(LEN(h$)-i)
400 END FOR i
410 RETurn decimal
420 END DEFine hex2dec
430 :
440 DEFine FuNction dec2hex$(i,flag)
450 LOCal low,high,result$,loop
460 high=i
470 result$=""
480 REPeat loop
490 low=(high/16-INT(high/16))*16
500 high=INT(high/16)
510 IF high=0 AND low=0 THEN EXIT loop
520 result$=hex$(low+1)&result$
530 END REPeat loop
540 REPeat loop
550 IF flag AND result$="" THEN result$="00"
560 IF flag AND LEN(result$)=1 THEN result$="0"&result$
570 IF flag THEN EXIT loop
580 IF LEN(result$)>=5 THEN EXIT loop
590 result$="0"&result$
600 END REPeat loop
610 RETurn result$
620 END DEFine dec2hex$
```

the four bytes of memory beginning at the symbolic address **'OPERAND1'.** This is necessary because the **'ADD'** instruction which follows requires that one of its operands be held in one of the data registers. What is this business about **(PC)** when referring to **OPERAND1?**

PC's Relatives

Instead of telling the computer to look for the operand at an absolute memory address, we are describing the data address in terms of a displacement from the current value of the program counter. This is known as 'PC relative' addressing and has the enormous advantage that programs which use this type of addressing are 'position independent' and will work correctly whenever they are placed in memory. You can test this by putting the program somewhere other than location 3FC00 – it will still work correctly! (NB – it must start at an even memory address or the machine will most probably hang up).

The **'ADD'** instruction is virtually self explanatory – it adds the source operand to what is contained in D1 and leaves the result in D1. The next job is to move the sum to the four bytes of memory starting at **'RESULT'.** Unfortunately, PC relative addressing is only permitted for source operands on the 68008 so we use the **'LEA'** (standing for 'Load Effective Address') instruction to make register A1 into a pointer to RESULT. Then we can move the data from D1 to RESULT using the 'address register indirect' mode and the program is still position independent.

Value Judgement

Finally, it is time to return control to SuperBasic. It is a convention with QL machine code programs that SuperBasic treats any value other than 0 in D0 as an indication that an error has occurred and will print an appropriate message. In this instance, there should have been no error so we use a variant of the **'MOVE'** instruction to ensure that D0 contains 0. The last instruction tells the processor to resume processing from the address after the instruction which called the routine – in this case SuperBasic.

The machine code routines in this series were prepared using the author's own Assembler and Debugger.

QL

SKETCH PAD	**£14.90** ★
PARALLEL PRINTER INTERFACE	**£34.90** ★
JOYSTICK ADAPTOR	**£ 5.90** ★

SKETCH PAD

A graphics design package allowing the user to create high resolution full colour pictures and diagrams using the cursor keys or a joystick. Features include:-

★ Screen dump to any Epson compatible printer.
★ Microdrive save and load of screens.
★ Use of stipple gives a vast range of colours.
★ Graphics and text can be freely mixed.
★ Automatic drawing of many shapes.
★ Filling of regular or irregular shapes in any colour.
★ Drawings saved on microdrive can be redisplayed from within any Super-Basic program using simple procedures supplied on microdrive.

Some applications include:-

★ Labels ★ Signs ★ Flow Charts
★ Graphics for Games
★ Artists Drawing

****

SPECIAL OFFER
SKETCHPAD PLUS JOYSTICK ADAPTOR £19.95

****

PARALLEL PRINTER INTERFACE

Simply plugs into the 'Ser 1' port of the QL and any centronics compatible printer (eg:- EPSON, CANON, OKI, Kaga, Juki etc.)
works with PSION software and requires no special commands. Full instructions included.

JOYSTICK ADAPTOR

Enables any Atari compatible joystick (eg: Kempston, Quickshot) to be used with the QL joystick ports.

All prices include V.A.T. and p & p within the U.K. Make cheques or postal orders payable to Sigma Research.

Sigma Research

231 Coldhams Lane, Cambridge CB1 3HY.

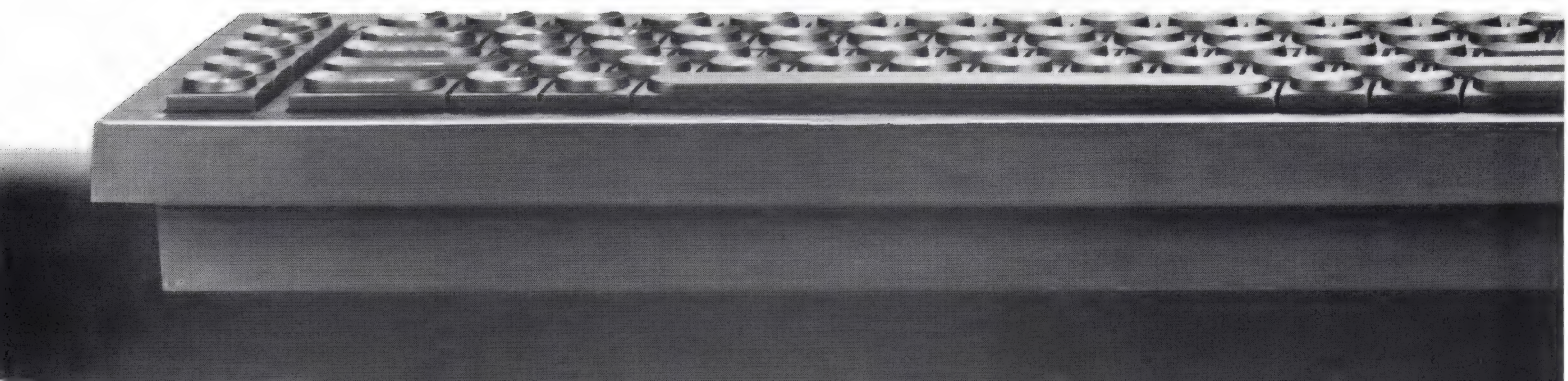
Recommended by Sinclair Research Limited. Guaranteed, designed and manufactured by O.E. Ltd, Cumbria

This modular communications package gives your QL more versatility and potential than probably any other add-on.

Using expansion connectors, the units stack on top of each other to provide a universal communications facility for VT100 emulation, telex, electronic mail, home banking and teleshopping, user-to-user communication, rapid data and word processor transfer between sites, downloadable software and access to the international PSS network and databases such as Prestel and Micronet 800.

The modules can be purchased individually, starting with the Q CON for interfacing, micro-based I/O control, and midrodrive software support for itself and subsequent modules. It accepts any modem, including of course the Q MOD V23 modem. Q CALL provides unattended auto-dial and answering.

Take a quantum leap into cor



QL SERIAL TO PARALLEL CONVERTER

Switchable baud rate. This is necessary because the baud rate on the two serial ports cannot be set independently, and unless you can set the baud rate on serial to parallel converters your printer will be inoperable while using modems and other communications devices.

Specifications are as follows:

- ★ BAUD RATE SWITCHABLE from 75 right through to 9600 baud
- ★ CRYSTAL CONTROLLED for reliable transfer of data
- ★ QDOS and SUPERBASIC compatible
- ★ Drives any CENTRONICS printer

★ You are not restricted to a set length of printer lead because we have fitted a six-way pcc socket to our converter (same as QL) so any length of lead can be fitted at any time.
★ Self-contained, colour matches the QL, full one year guarantee, comes complete with 1.5 metres of cable (for custom lead lengths see below).
For custom lead lengths please add £1.15 per extra metre of lead required.
Note also available soon: (serial to parallel) switchable to (parallel to serial) communications interface with selectable baud rate, parity, stop bits and data word lengths. Please phone for details. Open 9am-5pm Mon-Fri, 9am-4pm Sat.
Introductory price of only £43.00 ex VAT, £49.45 inc VAT.

Open
9am-5pm Mon-Fri
9am-4pm Sat

TV/MONITOR PHILIPS 14"
Converted to run specially for the QL. £247.25 inc VAT

DON'T BUY A COLOUR MONITOR FOR YOUR QL
HAVE YOUR 14" & 16" PHILIPS or PYE COLOUR TV
CONVERTED TO A TV/MONITOR
Normal TV £62.10 inc VAT
Remote Control £66.70 inc VAT
Teletext £72.45 inc VAT

CARE ELECTRONICS
TEL: 0923 777155

- ★ Resolution better than 585 × 450 pixels
- ★ Image clarity comparable to leading monitors
- ★ Includes RGB lead for connecting with QL
- ★ Conversions carried out at our workshops within 2/3 days
- ★ Please telephone for other makes

UNIT 14, PEERGLOW INDUSTRIAL ESTATE, OLD'S APPROACH, TOLPITS LANE, WATFORD, HERTS WD1 8SR. Telephone: 0923 777155

But before you leap at the likely cost of all this, just take a look at the price details in the order coupon. Because in the same way that the QL gives you so much for so little, the cost of communicating now takes a downward jump of the imagination.

nmunications.

Q MOD
V23 modem. Direct telephone line connection, 1200/75 baud full duplex operation for Prestel and other database access, 1200/1200 baud half-duplex operation for user-to-user communication.

Q CALL
Auto dial/answer unit for attended or unattended operation.

Q CON
Asynchronous multi-speed interface, single chip micro based with software on microdrive cartridge. Operates at 9600, 4800, 2400, 1200, 300 and 75 baud. Independent Tx and Rx rates. 25-way D-type, SER 2, power and expansion connectors. Provides VT100 emulation and software support for Q MOD and Q CALL. Can be used with any modem.

Tick	Quantity	Module	Price*	Value
☐	_____	Q CON	£79.95	_____
☐	_____	Q MOD	£79.95	_____
☐	_____	Q CALL	£49.95	_____
			Total order value	_____

☐ I enclose a cheque/P.O.† made payable to O.E. Ltd.
☐ Please charge my Visa/Access† card
Card No. ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Signed _____

☐ Please send me more information.

Name _____
Address _____
_____ Tel: _____

QL3

oel

O.E. Ltd, North Point, Gilwilly Industrial Estate, Penrith, Cumbria CA11 9BN. Tel: 0768 66748 Telex: 64157

*Inc. VAT and P&P. Please allow 28 days for delivery †Please delete as appropriate.

Printer Impressions

The second part of our extensive QL printer and interface round-up.

For the vast majority of personal computer users a printer constitutes the most desirable add-on for their system. With QL users this figure is likely to be very much higher as they seek to benefit from the bundled software that accompanies the machine. Most, however, will enter into the business of choosing a printer with the misplaced notion that the machine they are after is simply a glorified typewriter – nothing could be further from the truth.

Microprocessor controlled, the current generation of low-cost printers are similar to typewriters only in so far as both may be used to produce printed matter, or 'hard copy' as it is known in computerspeak. Beyond this printers with their advanced formatting capabilities, variable typefaces, character sets and print speeds have little in common and much more to offer.

Printers fall into three broad categories: thermal transfer, dot matrix and daisywheel. Thermal transfer printers tend to be "cheap and cheerful". The print mechanism either burns a character's image on specially treated paper or uses heat to transfer specially treated ink from a ribbon cassette onto plain paper. The principal advantage of this method is that transfer is relatively silent. The disadvantages are that the print quality is poor when plain paper is used and that thermal paper is expensive. They are, however, convenient and cheap.

Dot Versus Daisy

Dot matrix printers represent the next step up the scale where price and print quality go hand in hand. Here the print mechanism is an assembly containing a single column of seven or nine minute metal rods or pins, each of which may be fired out onto an ink ribbon to create a dot on paper. A single character is fashioned from a matrix of these dots in much the same way as one is printed out on screen. Obviously, the larger the matrix (the greater the number of pin positions within it) the more clear-cut the letter will be in print.

The beauty of a dot matrix printer is that there is no restriction in the shape of a character formed by the pins. This means that not only does the user have access to a wide range of preset typefaces stored in ROM but can also define his own characters and scientific symbols and print out graphics displays, such as those generated by Easel. In addition the user will benefit from much higher print speeds.

Whereas hard copy produced by this method is easily legible, its 'dotty' appearance renders it unsuitable for business correspondence and other areas where similar stringent standards of presentation apply. Nevertheless on the latest generation of dot matrix printers a number of refinements ('double strike' mode, 'emphasised print styles' and Near Letter Quality type) have been incorporated which reduce white space between the dots to give the kind of "solid" style type suitable for most correspondence. To test this facility, users should print out a document using high quality paper and then check for the presence of true descenders and ascenders by looking closely at such letters or symbols such as: M, 4, 8, @, $ and &. A final point, print speeds drop as the dot density of the typeface increases. For example, a speed of 160 cps for normal print will be reduced to 25 cps when NLQ is used.

Daisywheel printers make up the third and last category. The print mechanism is similar to that on the most advanced typewriters. Characters are embossed on the spokes of a light plastic wheel. The spokes number between 96 and 128. To print a character the wheel is rotated at high speed until the desired character is aligned with a print hammer. The hammer then strikes the character onto a ribbon. As the mechanism ensures accurate positioning and uniform striking, print quality is superb.

Aside from cost, daisywheel printers are noisier, slower and less versatile than their dot matrix counterparts. Typefaces may be changed by simply slotting in a fresh daisywheel, however, neither graphic displays nor definable characters are supported.

By now it should be apparent that selection of a printer from one of the

three categories listed will depend very much on the uses to which a QL is put. For purely educational/ recreational applications the very cheap thermal printer should be sufficient as it will produce listings to aid in program development plus the odd document or memo. If on the other hand, the QL is to be used primarily for word-processing within a business environment, then a letter quality daisywheel printer would be your first choice. Finally, a fast dot matrix printer is essential for serious applications which regularly require production of large quantities of information such as monthly invoices, listings, management reports, internal memos and sales forecasts.

Interfaces In Print

Whilst interfacing has been covered in a previous issue it is worth pointing out that few printers will connect up directly to the QL. Most require a serial to parallel interface. Fortunately, a number of easy-to-use plug-in QL interfaces are available costing between £30 and £50.

Affordable Portable

The brother HR5, capable of generating an 80 column text and supporting a full ASCII character set, plus 63 graphic blocks, is one of the smallest "full feature" printers on the market. Using a thermal print mechanism, the HR5, though cheap and silent in its operation does have its limitations.

Whilst the machine will print to paper using a special ribbon cassette, print quality is of a lower standard than that obtained on most impact dot matrix printers. Legibility, however, can be improved either by using more expensive thermal paper or else smooth glossy paper. Photocopy paper is ideal.

Highly welcome, the HR5 comes with a built-in RS232C interface which permits direct connection with

PHOTO TERRY BEDDIS

Choosing a printer is a lot easier than choosing a computer.

THERE are dozens of quality printers from which to choose. With quality price tags of around £250.

The Brother M-1009, however, breaks all the rules.

Stays defiantly below the £200 barrier.

Though it has far more than its fair share of features, it maintains the extraordinarily low price of £199.95.

Travels at a steady fifty.

In the speed stakes, the M-1009 is certainly no slouch, being fully capable of up to 50 characters per second.

Providing bi-directional and logic seeking printing for normal characters and uni-directional printing for super and sub script and graphics.

Prints on any paper.

Being an impact printer, the M-1009 will print on virtually any paper, including letter headings, invoices and standard office stationery.

It will even print two copies together with your original.

A superb character recommendation.

In its price range, the M-1009 has a great deal more character than many printers.

96 no less, plus international type and graphic characters.

Reliability comes as standard.

Built to the same exacting standards as Brother's elite office printers, the Brother M-1009 already has faultless credentials for reliability.

Its 9 pin dot matrix head, for example, has an astonishing 20 million character service life.

One printer that doesn't block out the light.

Many home computers tend to be a little on the large side.

In contrast, the compact M-1009, at only 7 cm high, keeps a discreet profile.

Well designed, reliable – and conscientious.

The Brother M-1009.

brother®

The future at your fingertips.

DEPT P, BROTHER OFFICE EQUIPMENT DIVISION, JONES + BROTHER, SHEPLEY STREET, GUIDE BRIDGE, AUDENSHAW, MANCHESTER M34 5JD.
TEL: 061-330 6531 (10 LINES) 061-330 0111 (6 LINES) 061-330 3036 (4 LINES) TELEX: 669092
BROTHER INDUSTRIES LIMITED, NAGOYA, JAPAN.

AVAILABLE FROM
BOOTS, WILDINGS, SPECTRUM, JOHN MENZIES, MICRO MANAGEMENT, MAJOR DEPARTMENT STORES AND ALL GOOD COMPUTER SHOPS.

the QL. Unfortunately, we understand that the printer lead supplied with the QL cannot be used with the HR5 and that users will have to purchase a special lead from Brother themselves.

The HR5 runs off four UM-1 batteries or an optional 6V DC mains adaptor. The printer has a power-off switch but no auto power-off facility for when it is running off batteries.

Small, compact and battery operated, the HR5 is the perfect candidate for those who want a second, portable printer to complement their QL on site, location or on the move.

A Legendary Workhorse

The Epson's RX 80 like its more expensive stablemates is a sturdy dot matrix printer built to last. As its name implies the printer normally prints out 80 characters on a line, but this can be increased up to 137 depending on the print style chosen.

An 8K built-in ROM provides the RX 80 with a broad range of formatting commands, alternative typefaces and character sets. These make the printer an extremely versatile and sophisticated tool. For example, 'emphasised' print style may be combined with 'double strike' mode to conceal dots and give text a more 'solid appearance'.

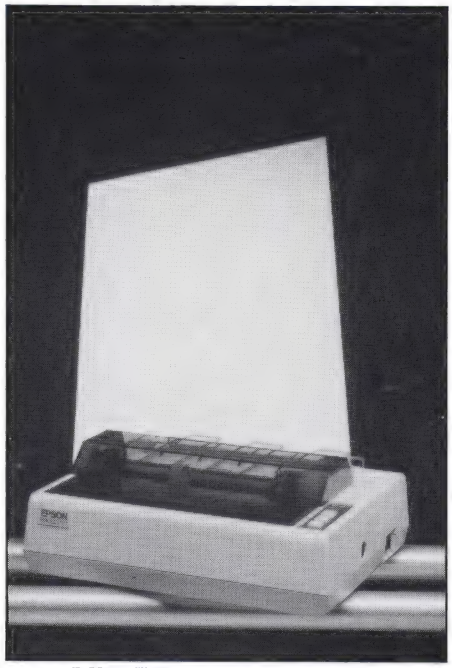
In addition, the precision of the RX's printout makes it exceptional for producing screen dumps and ideally suited for use with Psion's Easel.

In operation the RX 80 is quite noisy, however, there is a facility to reduce print speeds to 50 characters per second which results in a 3 Db reduction in noise levels. Changing typefaces reduces print speeds as well.

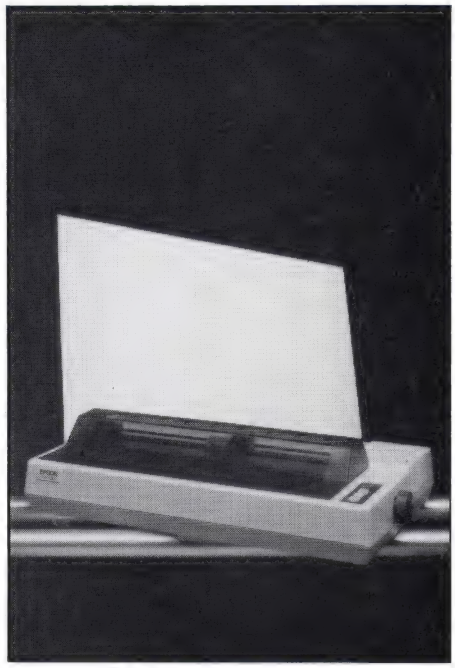
As with most printers, the RX 80 requires a serial to parallel interface to connect up to the QL. Epson themselves along with Micro Control Systems provide one to fit inside the printer. Installation, whilst not straightforward is carefully outlined in the handbooks accompanying the devices. A broad range of print buffers are also available.

The F/T version includes both friction and tractor feed and will accommodate continuous stationary from 4″ to 10″ wide and single sheets up to a maximum of 12″ wide. Also included is a paper path attachment which ensures that fan fold paper is correctly aligned and so prevents it from being chewed up by the sprockets on the tractor feed. One minor drawback, however, changing paper requires considerable dexterity. An optional roll paper holder is also available.

Along with its faster cousin the FX 80 F/T, the RX is an exceedingly popular choice amongst personal computer users. With its low cost, almost legendary reliability and a large repertoire of functions fast becoming an industry standard it is not difficult to see why this is the case.



RX80 F/T remains ever popular.



FX-100: 132 column text.

Over Its Prime

The FX 100 is Epson's top-of-the-range printer and offers all the features associated with the RX, including an excellent manual which carefully details every facility and provides graphic examples and test programs in all cases.

The FX produces 136 columns of text in normal type. Again, this may be increased to a maximum of 223 in condensed type. While most will be content with an 80 column printout, the extra capability may appeal to those who use Psion's Abacus to generate large spreadsheets. Maximum paper size is 16in. for continuous stationery. The FX 100 does not support single sheet paper though an F/T option is available.

Epson's FX series differs in a number of important respects from the RX. First, the dot density of the matrix is greater, permitting sharper character definition in all print modes. This has also been reflected in marked improvements in the graphics capabilities where three additional print modes are available including one for graph plotters. Print speeds have also been increased by 60% across the board.

Next, the FX series contains a 2K print buffer which can either be used to speed up printing or to store user-defined characters. Additionally, Epson have added a proportional print mode to their established repetoire of functions.

On the design side, Epson have made the FX's DIP switches easily accessible by incorporating a small removable panel on the side of their machines through which they may be reached. This saves having to dismantle the unit to lay bare its printed circuit board, a tedious and time consuming course of action required by RX owners.

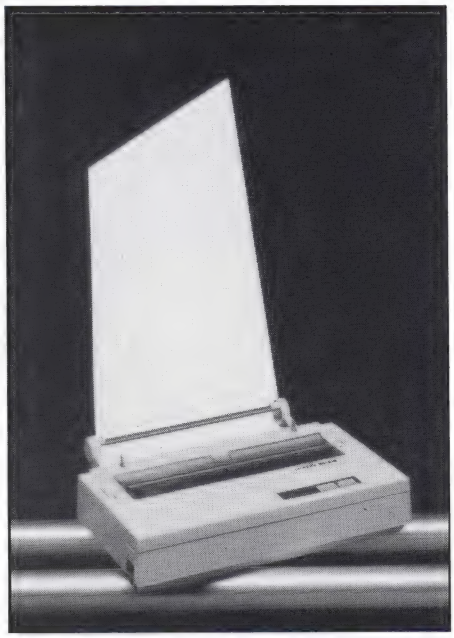
With its increased speed and additional typefaces the FX series would appear to be best suited to business applications. However, bearing in mind the higher prices that FX 80 and FX 100 command, it seems strange that neither incorporate some form of NLQ print mode. And this omission makes the FX series somewhat dated in relation to its rivals.

A Familiar Name

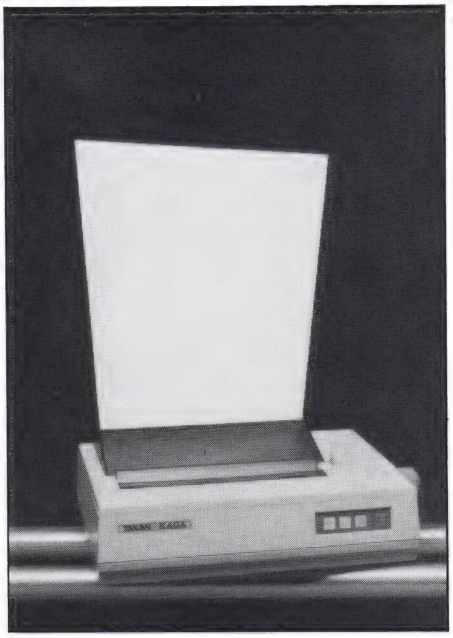
Taxan/Kaga will already be known to most QL Users as their 12in. QL Vision monitor is endorsed by Sinclair and likely to have found its way into many a workstation. Their KP-810 printer looks set to go the same way even without the Sinclair tag.

The KP-810 combines all the advanced features of the latest generation of dot-matrix printers into a single low cost package. It supports 80 columns per line and accepts cut sheet, fanfold and roll paper. In the last case this is achieved by simply designing the paper feed area in such a way that with the cover down, the roll nestles in a trough and is able to rotate freely without the need for any special holding mechanism.

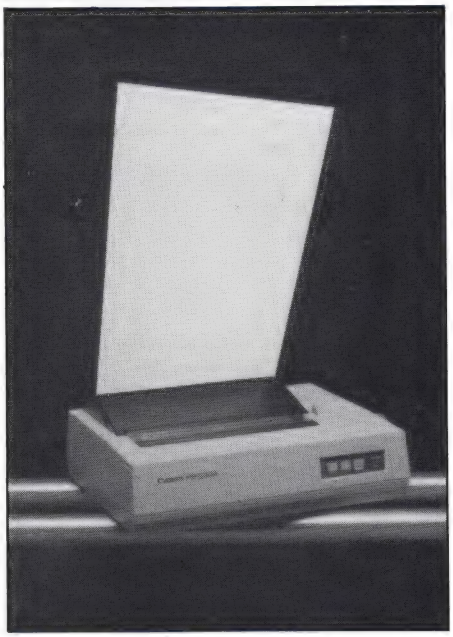
The KP's repetoire of typefaces is equal to that of any other dot matrix currently available. In addition, printer control codes are Epson compatible. As an Epson printer driver is standard on Quill this means that QL users should have little or no trouble in producing hard copy for any one of Psion's programs, including Easel.



HR-5: Battery powered portable.



Taxan-Kaga's NLQ printer.



Canon's faster clone.

As with more expensive printers, the KP-810 incorporates a 2K print buffer and a 3K input buffer. The former speeds up printing. The latter may be used either to reduce the amount of time your QL remains "printer bound" or to contain up to 256 user-defined (PCG) characters. Selection between the two options is made by setting a DIP switch.

Perhaps the most attractive feature of the KP-810 is its ability to support an NLQ mode. In this mode, the matrix dot density is effectively doubled up from 11 × 9 to 23 × 18 giving very good quality print, though admitedly at a greatly reduced print speed (27cps).

On the design side the KP-810 is functional rather than attractive. The controls are located on the front side of the machine, and with the power switch on the right hand side. Both are easily accessible. The same cannot be said of the DIP switches which require the unit's casing to be removed in order to gain access to the PCB where they are located.

All in all, at £300 the KP-810 with its extended print modes and high density dot matrix configuration is an extremely strong contender. The promise of Near Letter Quality print, coupled with Epson compatibility makes it a natural choice for QL users in search for versatility and quality but on a limited budget.

Mistaken Identity

The Canon PW-1080A dot matrix printer bears more than a passing resemblance to the Taxan KP-810 (or for that matter the Centronics Horizon HP-80). Both support an identical repetoire of print styles, characters sets and formatting functions including NLQ mode. Furthermore, aside from a different logo, the casing is identical right down to the off-white colour scheme (very tasteful!)

In fact, the only difference between the two, at least on paper, is a 20cps increase in print speed which puts the Canon amongst the fastest of the low cost printers currently available. In operation, however, this increase was hardly noticeable with the Canon performing no better or worse than its cheaper rival.

As with its look-alike, the Canon supports 80 column type and accepts cut sheet, fanfold and roll paper. The unit permits five different settings for paper density to accommodate carbon copies and multipart stationery. Furthermore, its membrane control panel is set on the front of the machine and need not be disconnected when dismantling the unit which makes it much easier to set the DIP switches located internally.

The operation manual included with the printer was one of the few supplied without an errata sheet. Well-laid out, it was comprehensive in its coverage with no shortage of test programs, character tables and circuit diagrams. It should appeal to beginners and experts alike, permitting both to get the most out of their printer.

Finally, the printer affords a number of expansion options. These include additional NLQ ROMs offering Italic, Gothic, Orator and Courier typefaces. Also available is a PCG RAM extension that provides the printer with a maximum of 8K for user-defined characters and a serial interface board.

To sum up, the PW-108A would appear to be an "up-market" version of a winning, if not original, combination. Marginally more expensive, users will be paying a premium for the name as well as the increase in speed. Nevertheless, along with other NLQ printers it represents not only stiff competition for better established dot matrix printers but also a cheap and viable alternative to a letter quality. daisywheel printer.

Business Thoroughbred

The Canon PW-1156A was the largest and heaviest printer reviewed. It is identical to the PW-1080A in every respect, aside from the fact that it will print up to 156 characters in normal text or 265 in condensed, on paper up up to 17in. wide. This capability puts it into the same league as the Epson FX-100. Here, with its additional NLQ print style and competitive price the printer enjoys a distinct advantage.

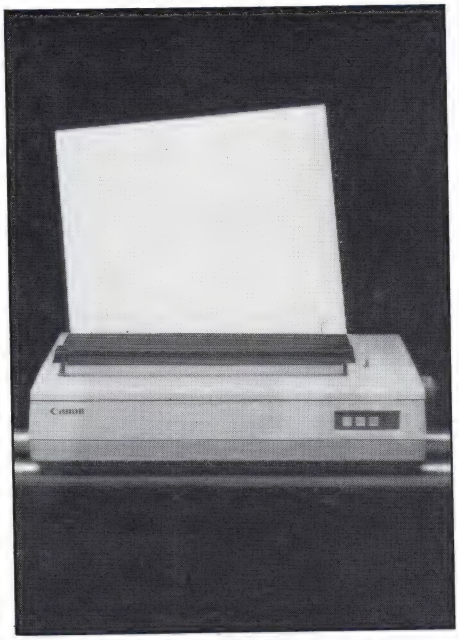
Whether the average user can justify such a printer depends on how he intends to put his QL to work. The PW-1156A is most suited to business or scientific applications which require the use of spreadsheets or similar packages that generate large amounts of information in tabular form. In this instance, the permittable increase in paper widths will dramatically improve legibility of such reports.

Daisy Daisy

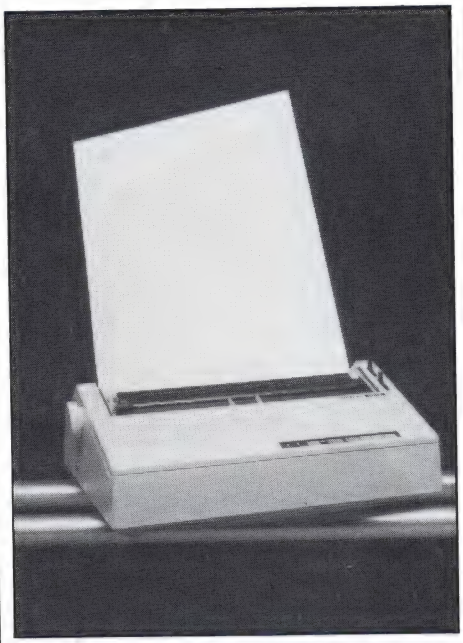
Brother would appear to be fast acquiring a reputation for breaking price barriers. Their dot matrix printers and electric typewriters are generally more innovative and considerably cheaper than existing equivalents. The same cannot be said of their daisywheel printers.

The HR-15 supports a maximum of 165 characters per line and prints onto cut sheets (max width 13.5in). The 15cps print speed is slow but more expensive models are available that will boost it up to 35cps.

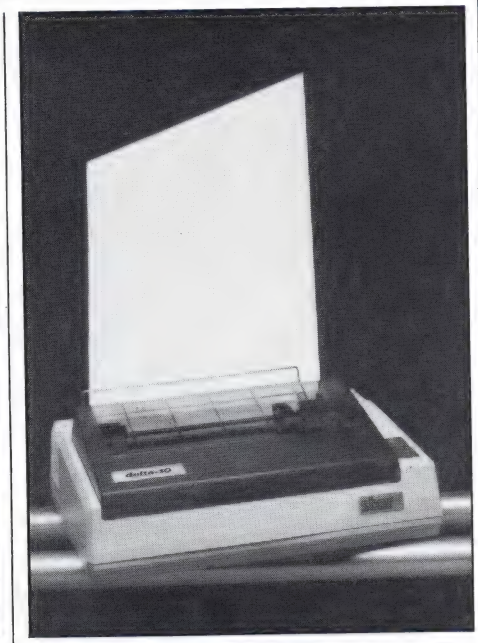
In common with most daisywheel printers, the HR-15 offers two colours (red/black) as well as shadow print-



1156A — Full width plus NLQ.



Brother's HR-15 daisywheel.



Delta-10: Tried and tested.

ing, automatic underline, proportional spacing, super and sub script. A particularly useful bonus is a 3K internal print buffer which may be used to store a document. This allows you to free the computer for other tasks whilst printing copies.

Print quality is excellent which is as it should be with a letter quality printer. In operation, the machine tends to clatter out at a noisy but certainly not intolerable level.

In its design, the printer is compact and well finished with six membrane controls and an array of indication lights set flush in a switch panel on top on the machine.

Also, to their credit Brother have seen fit to make the DIP switches easily accessible by locating them alongside the various input/output ports at the rear of the machine. The switches are split into two groupings of 8, named SPECs 1 and 2. The former group deal with differing printerwheel types and page lengths. The latter determines line feeds, parity and Baud rate. Parity may be even or odd, Baud rates range between 110-9600 making its particularly easy to connect up to the QL.

Rising Star

Like Brother, Star produce a broad range of low cost printers of which the Delta-10 occupies the pole position. Unlike the competition, however, Star have tended to consolidate where others innovate. As a result whilst no single feature on their printer is exceptional, the number included is.

An 80 coloumn dot matrix printer, the Delta 10 comes with both RS-232C and parallel ports. Though only one may be used at any one time, switching between the two has been made particularly simple by exposing the relevant DIP switch at the rear of the machine. This spares the user the effort of dissembling the printer each time he wishes to alternate ports, he need only simply turn the machine about and flick the second switch "on" (serial) or "off" (parallel).

As a further bonus, the Delta-10 includes a built-in 8K input buffer which dramatically reduces the amount of time the computer is "printer bound". On many other printers such a facility is available, though only at a price. Tractor feed and roll paper holder are also included.

Extras aside, the Delta-10 offers a fairly standard array of print functions, character sets and typefaces. Also head size is less than that obtainable on other printers within the 160 cps category.

Functional in design, the printer's controls are located up front and are of the push button variety, as opposed to membrane. But far less attractive is the fact that the unit employs a spool type arrangement for the ink ribbon similar to that on a typewriter. This makes ribbon replacement a messy and protracted affair when compared to the "plug and go" cartridges. However, this may be mitigated by the fact that spools are cheap and easily obtainable.

Finally, the "preliminary" users manual, though adequate in parts, provides insufficient information regarding control codes. Thus only the most experienced user will be able to benefit from many of the printer's functions.

Overall then, Star's Delta printer should appeal to those where speed and ease of connection are more important than print quality. Certainly, with its built in buffer it is ideally suited to producing listing without tying up the printer for long spells. However, where documents are concerned a NLQ printer might well be the preferred choice.

Spirited Rival

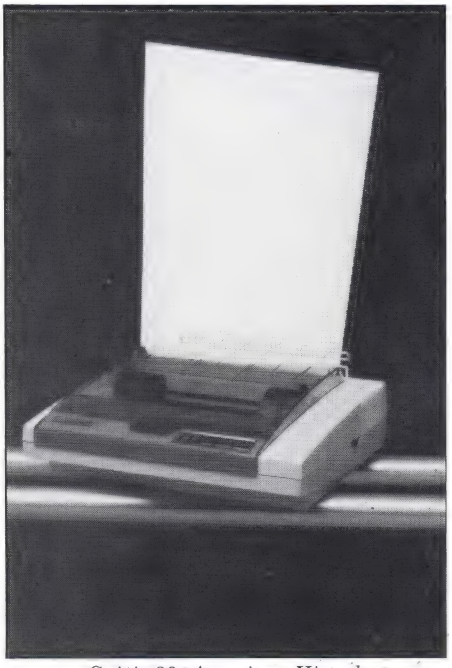
Pioneers of the dot matrix printer, Mannesman Tally should have plenty of experience to draw on and superficially, at least, their Spirit 80 low cost printer reflects this. Aimed as a competitor for Epson's RX 80, the Spirit 80 in its sleek, hi-tech casing streaks ahead of the opposition. Looks alone, however, do not make a printer.

To begin with, the print head size of the Spirit 80 is inferior to all the printers reviewed. This means that print quality is less than average and also explains the sizeable reduction in bit-image modes and resolutions. In the latter case this is unlikely to affect the printer's capacity to produce Easel graphics, but may well prove to be a hurdle if more sophisticated programs are developed.

Next, prints speeds whilst adequate still fall short of those obtainable on equivalent printers. As regards print styles, the Spirit 80's complement seems fairly standard though it omits proportional and NLQ modes.

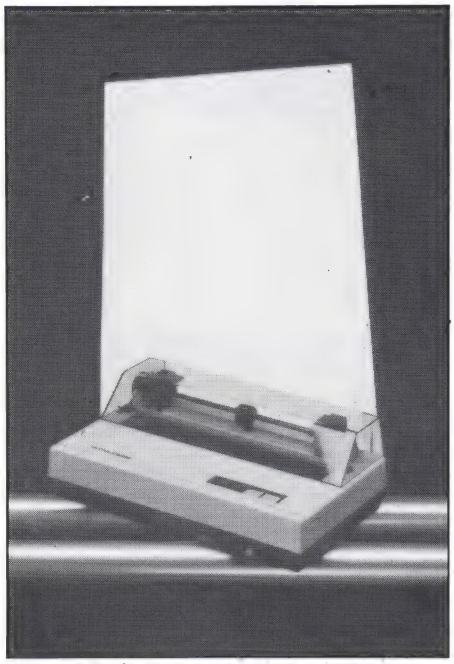
Moving on, the MT 80 is capable of supporting plain and continuous stationery. The tractor feed cannot be detached, though it is neatly recessed to permit single sheet paper to pass over it without obstruction. Paper is fed out through a slot in the lid. This has a serrated edge giving the printer what blurb described as "unique quick tear facility". The principle closely resembles that of kitchen foil dispensers.

In operation, the printer was quieter than most and as control codes were ASCII and Epson compatible, few problems were encountered in generating documents from Quill. Though had there been any, it is unlikely that the short operator's



Spirit-80: American Hi-tech.

Ensign's not quite NLQ printer.



M-1009: Latest low cost Brother.

manual would have provided much guidance.

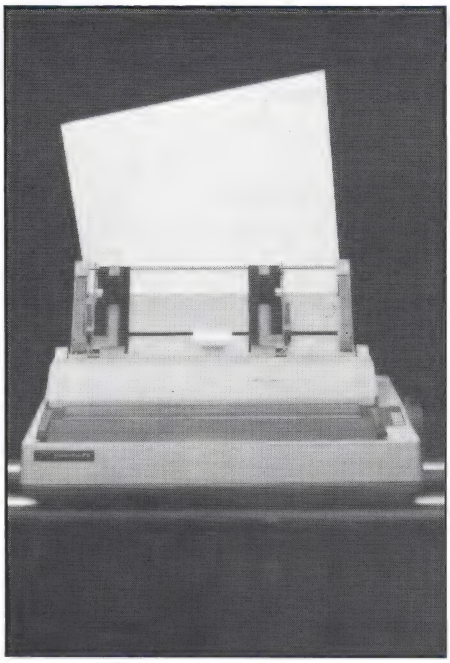
Hoisting The Ensign

Fastest amongst the printers reviewed, the Ensign 1650 is a recent introduction from Japan. The 80 column printer supports single sheet and fanfold paper and sports a few unusual features.

First, along with the usual assortment of print styles including proportional, the Ensign adds one of its own. Known as "high density" type, this is a toned-down version of NLQ which takes into account the printers reduced head size. Here letters are formed over a 17 × 17 dot matrix as opposed to the 23 × 18 for NLQ proper. Nevertheless print quality is of a high standard and puts the printer into a class of its own.

Second, in its design the 1650 departs from the usual formula in a number of ways. The lid is split into two parts, with one part ostensibly for noise reduction. No paper guide has been included, which is a shame as fanfold paper can often twist and get chewed up in the tractor feed. On a brighter note, one of the printer's two DIP switches is easily accessible at the rear of the machine. This will permit the user, for example, to reserve the 2K input buffer for user defined "download" characters without having to dismantle the machine. Also, the parallel port is set vertically and well to the left at the rear, ensuring that the cable does not interfere with paper feed.

Make	Brother	Brother	Mannesmann Tally	Epson	Quen Data	Taxan/Kaga
Model	HR-5	M-1009	Spirit 80	RX-80 F/T	1120	KP-810
Price inc VAT	157.00	224.00	250.00	290.00	295.00	366.85
Type	Dot Matrix	Dot Matrix	Dot Matrix	Dot Matrix	Daisywheel	Dot Matrix
Print method	Thermal Transfer	Impact	Impact	Impact	Hammer	Impact
Head Size	9 × 9	9 × 9	7 × 8	9 × 9	N/A	11 × 9
Characters/sec	30	50	80	100	18	140
Pica Character/line	80	80	80	80	180	80
Dimensions	300 × 170 × 70	330 × 190 × 70	380 × 295 × 125	360 × 300 × 125	500 × 330 × 160	400 × 320 × 110
Weight	0.5 Kg	3.0 Kg	5.3 Kg	7.5 Kg	9.5 Kg	8 Kg
Power	6 Volt DC	Mains	Mains	Mains	Mains	Mains
Print Modes	P, EN, EL, C, C-EN, SUB, SUP	P, EN, C, C-EN SUP, SUB, DS, EM	P, EN, C, C-EN DS, EM, SUB, SUP C-EN. EL-EN	P, EN, EM, EL C, C-EN, EL-EN DS, SUB, SUP	N/A	P, EN, C, C-EN EL, EL-EN, NLQ DS, SUP, SUB NLQ-EN, NLQ-PR
ASCII Characters	96	96	96	96	96	96
Ital Characters	None	None	96	96	N/A	96
Graphics Chars	64	64	32	32	N/A	256 (def)
Int Character sets	—	—	8	11	N/A	8
Bit Image Modes	2	4	2	6	N/A	11
Dots/inch	120	240	120	240	N/A	240
Max Dot Mode	8	8	8	9	N/A	16
Interface	RS232C	Centronics	Centronics	Centronics	RS232C	Centronics
Print Buffer	—	—	—	—	—	2K
Paper	Thermal/Plain	Plain	Plain	Plain	Plain	Plain
Paper Feed	Friction	Friction	Tractor/friction	Friction/tractor	Friction	Tractor/friction
Carbon Copies	—	—	3	2	3	2
Ribbon Type	Special Cassette	Special Cassette	Cartridge	Cartridge	Cassette	Cartridge
Ribbon life/char	50,000	½ million	3 million	3 million	—	3 million
EXTRAS	Mains adaptor	RS232C, Tractor Feed, Roll paper holder	RS232C	RS232C	Tractor Feed, Cut Sheet Feed	RS232C, NLQ Fonts
SUPPLIER	Thame Systems, Thame Park Rd, Oxon	Quest International, School Lane, Chandler's Ford, Hants	RAM Computer Centre, 117 Queensway, Bletchley, Bucks	Strong Comp Systems, Bryn Cottage, Peniel, Camarthen, Dyfed	RAM Computer Centre, 117 Queensway, Bletchley, Bucks	Twickenham Computer Centre, 72 Heath Rd, Twickenham, Middx



Sheet feeder attachment.

Finally, the manual supplied with the printer was one of the best in terms of presentation and clarity. All aspects of the printer's operation were meticulously detailed with different print styles well illustrated with sample programs, printouts and diagrams showing different dot densities.

With much to offer, the Ensign would be an attractive proposition, were it not for a price tag that puts it in the same league as fully fledged NLQ printers.

Mighty Quen

When it comes to producing printed matter of 'letter' quality, daisywheel printers are prime candidates. However, compared with most of the available alternatives, the price is considerable and coupled with slow speed they would appear to present a poor case compared to dot matrix printers.

The Quen Data 1120 changes all that. Here is a daisywheel printer of reasonable speed and quality, but at a price that's well within the reach of most QL owners (around £295).

The 1120 provokes a feeling of dèja vu, for those who are familiar with daisywheel printers, as in another guise it appeared as the Daisy Step 2000. However, for those unfamiliar with this type of printer, the basic mechanism is a small hammer which strikes a rotating wheel with the characters positioned at its edges. The action is rather like a typewriter except that instead of individual hammers for each key, there's a single hammer with all the characters mounted on a single wheel (the daisywheel).

All the controls are positioned on the front of the machine rather than on the top, as with the aforementioned Daisy Step. This is handy since it leaves space for the DIP switches just inside the lid. Here they can be easily adjusted (something QL owners will get used to if they're utilising several printers).

The printer controls allow switching from On to Off Line, page and line feeds as well as operating a character test mode – particularly useful if you want to try out a range of wheels.

When the printer is connected to the QL – serial interface to SER1 via the lead supplied (by Sinclair) – then comes the tricky bit of setting the DIP switches and typing the various control commands that allow the computer to talk to the printer. The settings we've used (see table) are not necessarily the only ones, but they work (!).

When using the QL on its own (not Quill), typing something like **open #5,serlhc** will get things moving (as long as you've set the Baud rate to 300). Inside Quill you only need to make sure that the correct printer driver has been installed.

The Quen data 1120, in operation, is exactly what you'd expect from a daisywheel – high quality printing at a fairly low speed. The noise level is not too disturbing, though we have heard quieter. The major plus-point, however, must be versatility – a single wheel change can bring forth italics, pi fonts and a host of other designs (the wheel supplied has no pound sign, so you'll need to get another one anyway).

A New Brother

If you're looking for a printer on a budget then the chances are it's going to be dot matrix (in fact the cheapest

Star	**Ensign**	**Canon**	**Brother**	**Epson**	**Canon**
Delta-10	1650	1080-A	HR-15	FX-100	PW-1156A
370.00	375.00	400.00	445.00	500.00	525.00
Dot Matrix	Dot Matrix	Dot Matrix	Daisywheel	Dot Matrix	Dot Matrix
Impact	Impact	Impact	Hammer	Impact	Impact
9 × 9	9 × 9	11 × 9	N/A	11 × 9	11 × 9
160	165	160	15	160	160
80	80	80	165	136	156
395 × 315 × 150	405 × 300 × 120	400 × 320 × 110	450 × 320 × 150	585 × 335 × 140	600 × 350 × 130
7.8 Kg	6.7 Kg	8 Kg	9 Kg	10.5 Kg	12 Kg
Mains	Mains	Mains	Mains	Mains	Mains
P, EN, C, DS EM, SUP, SUB EL, PR	P, EL, C, EN PR, DS, EM, SUP SUB, NLQ	P, EN, C, C-EN EL, EL-EN, NLQ DS, SUP, SUB NLQ-EN, NLQ-PR	N/A	P, EN, EM, EL C, C-EN, EL-EN DS, SUB, SUP, PR	P, EN, C, C-EN EL, EL-EN, NLQ DS, SUP, SUB NLQ-EN, NLQ-PR
96	96	96	96	96	96
96	96	96	N/A	96	96
32 + 192 (def)	256 (def)	256 (def)	N/A	256 (def)	256 (def)
8	8	8	N/A	9	8
4	6	11	N/A	9	11
240	240	240	N/A	240	240
8	16	16	N/A	9	16
Centronics + RS232C	Centronics	Centronics	RS232C	Centronics	Centronics
—	8K	2K	—	2.5K	2K
Plain	Plain	Plain	Plain	Plain	Plain
Tractor/Friction	Tractor/Friction	Tractor/Friction	Friction	Tractor	Tractor/Friction
2	2	2	4	2	2
Spool	Cassette	Cassette	Cassette	Cartridge	Cassette
—	3 Million	3 Million	—	3 Million	3 Million
—	—	RS232C, NLQ Fonts	Tractor Feed, Cut Sheet Feed	RS232C	RS232C, NLQ fonts, 8K PCG RAM Extension
Datastar Systems, 182 Royal College St, London NW1	Twickenham Computer Centre, 72 Heath Rd, Twickenham, Middx	Microperipherals, 69 The Street, Basingstoke, Hampshire	Relative Marketing, 182 Royal College St, London NW1	Printerland, Unit 27, Estate Buildings, Railway St, Huddersfield	Twickenham Computer Centre, 72 Heath Rd, Twickenham, Middx

alternative we've come across is the daisy wheel reviewed in this article). And looking in the sub-£200 region there is very little choice – the Brother M1009 being one of the few options.

The M1009 is a small neat printer, along the lines of the popular HR5, but with a built-in mains supply so there's no need to worry about batteries constantly running out.

The basic unit takes single sheets, though there is a tractor option as well as the slightly cheaper roll arrangement. Since the printer is an impact rather than thermal type (like the HR5, for instance), it requires a ribbon to be fitted, though this does not pose any problems as one is supplied with the machine.

As always the manual supplied with this Brother printer is of high quality, though we were rather amused at the accompanying errata, which is larger!

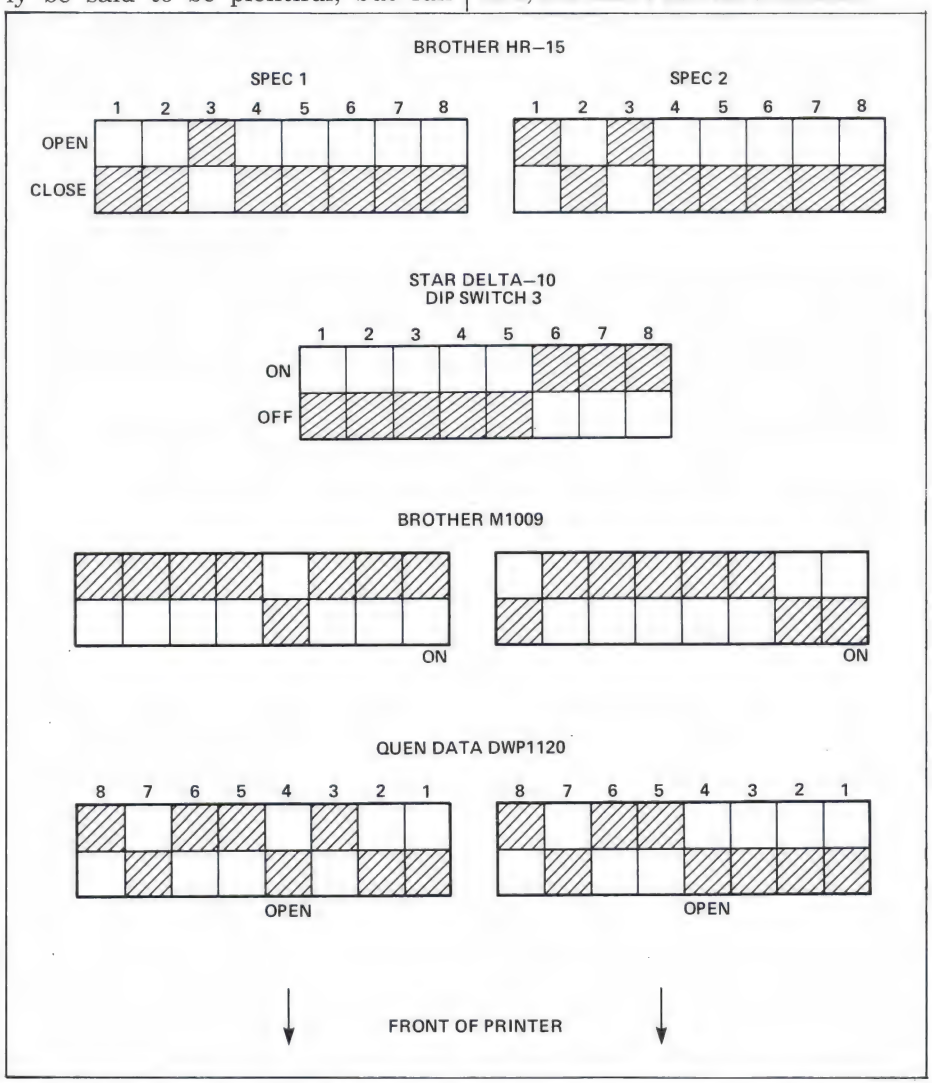
All the various setting-up operations are well described with clear diagrams. The DIP switch positions and their meanings are all given, though here is another printer where no examples are provided for specific computer settings – what a shame.

Controls on the M1009 could hardly be said to be plentiful, but full marks to Brother for putting the on/off switch on the front – there's nothing more annoying than connecting up the machine and mounting it in one of those tight-fit spaces only to find that you can't switch it on and off.

In common with most other printers there are a range of extended controls which can be accessed via the ESC sequence. In the case of the Brother M1009, these include a few unusual provisions for a printer in this price range (enlarged and condensed characters, double-strike mode, line spacing and the inclusion of bit-image mode – creation of unique characters is a definite possibility).

Brother are to be commended for the way in which each extended code is defined and an example given to show what they're talking about (other manufacturers take note). Not only that but each example includes a suitable listing for trying out the various features – modifications are required to match SuperBasic. Indeed, anyone considering the purchase of this printer could do a lot worse than read through the manual.

In use the M1009 is 'quick and quiet' with only its size and level of sophistication being limiting factors, but that's no real criticism.



DIP switch settings for serial printers (QUILL operation shown).

List of Abbreviations

(C) Condensed
This print mode reduces character width by half.

(DS) Double-Strike
Similar to emphasised mode, double strike "blurs" a character along the vertical, thereby eliminating the gaps between the dots on that plane. Again print speeds are reduced by half. Usually, double strike mode may be combined with emphasised to produce a gap free image. However, print speeds in this instance will be reduced by 75%

(EL) Elite
Reduces character width by a third.

(EM) Emphasised
Here dots are reprinted a fraction to the right of the original position. This sideways "blurring" action eliminates the horizontal gap between dots and increases legibility giving a character a more solid appearance. As a character must in effect, be printed twice over, print speeds are halved.

(EN) Enlarged
Doubles character width.

(NLQ) Near Letter Quality
In NLQ mode special characters are printed with virtually no blank spaces between the dots. This is achieved not by "blurring" a normal character but by using two passes to print out a single character whose shape is definable on a matrix twice the normal size. Perhaps, the best way of visualising this process is to see a single NLQ character as being made up of two impressions, one containing all information in the odd rows and columns of the matrix, the other all the even. In its first pass, the printer prints one impression, then drops a minute amount and moves on to print the second.

(P) Pica
This measure describes the normal print where maximum print speeds quoted will apply. Upon powering up, the printer will default to this unless DIP switch settings have been altered.

(PR) Proportional
This print mode enables one to reduce the amount of blank space that surrounds some characters. The overall effect would be to make the gap between letters and user defined characters equal no matter what their size. For example the gap between "ii" and "mm" would be the same. It is achieved by reducing the distance that the print head will travel after printing the character and is usually found on up-market word processors as well.

PRINTERLAND

ALL ORDERS DESPATCHED SAME DAY
SELLING PRINTERS SINCE 1980 !

FREE With all Printers. QL Users Printer Guide
Getting the best from your printer is NOT easy !
We supply a FREE booklet with all printers which explains how to obtain all the features available on your printer. Without this you could waste many hours. Buy elsewhere and you'll pay far more and get far less !

PRINTERS
DOT-MATRIX

	Ex VAT	Inc VAT
EPSON RX 80	£185.00	£212.75
EPSON RX 80 F/T +	£212.00	£244.80
EPSON FX 80 F/T	£314.00	£361.10
EPSON FX100 F/T	£425.00	£488.75

DOT MATRIX PLUS NEAR LETTER QUALITY

KAGA TAXAN 810	£259.00	£297.85
CANON PW1080A	£269.00	£309.35

DAISY WHEEL

JUKI 6100	£325.00	£373.75
BROTHER HR15	£329.00	£378.35
EPSON DX100	£356.00	£409.40

QL RS232 INTERFACES

MIRACLE SYSTEMS	£32.50	£37.37
EPSON RS232	£27.00	£31.05

WANTED
QL Software or Hardware
First Class Terms Offered

SHOWROOM OPEN 9.00 am to 6.00 pm DAILY. 9.00 am to 1.00 pm SATURDAY
DELIVERY PRINTERS/MONITORS (SECURICORP) **£10.00** Other **£1.00**

	Ex VAT	Inc VAT
SINCLAIR QL	£334.00	£384.10

DISK DRIVES AND QL BOOKS NOW IN STOCK

MEMORY EXPANSION BOARDS

64K	£82.00	£94.30
256K	£199.00	£228.85

SOFTWARE

METACOMCO ASSEMBLER	£49.00	£56.35
PSION CHESS	£15.60	£17.90

MONITORS

MICROVITEC CUB 653	£220.00	£253.00

FULL PACKAGE OFFER
Sinclair QL. Microvitec Cub 653
Epson RX80 F/T. Price includes all leads and interfaces. (Other printers also available)

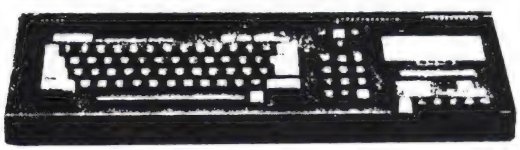
REC RETAIL (inc VAT)	£1070.00
OUR PRICE (inc VAT)	£899.00

Printerland, Unit 27, Estate Buildings, Railway Street, Huddersfield HD1 1JP
Telephone Huddersfield 514105/687875

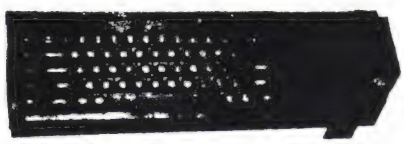
TWICKENHAM COMPUTER CENTRE LTD

With the best microcomputers available

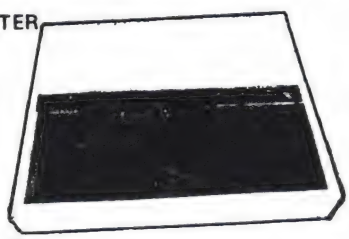
AMSTRAD CPC464



SINCLAIR QL



BBC MICROCOMPUTER



32K
30 lines of 80 characters
Serial and parallel output, analogue input, printer BUS connection

Quotations & expert advice. Full after-sales service

BBC Microcomputer

apricot

Plus software, peripherals, games, books and much more always available from:

Twickenham Computer Centre
72 Heath Rd Twickenham Middx TW1 4BW 01-891 4991

Nationwide delivery by Securicor

A HIGH RESOLUTION MONITOR FOR THE SINCLAIR QL AT A LOWLY £299.95 (INC.VAT).

This is the much acclaimed JVC colour monitor from Opus.

And it gives Sinclair QL owners a lot more than just excellent high resolution.

As you know, the QL gives out information in 85 characters on the screen instead of the standard 80 characters.

And the loss of those five characters means that ordinary monitors simply can't show you the whole picture.

Opus, however, have overcome this problem by adapting the JVC, via a special interface and cable, so that the QL can now work 100% efficiently on it.

All for the lowly price of just £299.95 including VAT, carriage and, of course, a year's full guarantee.

You can order it by post by filling in the coupon below and sending it to: Opus Supplies Ltd., 158 Camberwell Road, London SE5 0EE. (You'll receive it within ten days by courier service.) Or you can telephone 01-701 8668 and pay by credit card.

Alternatively, you can buy at our showroom between 9.00 – 6.00pm Monday to Friday, or 9.00 – 1.30pm Saturday.

MODEL REFERENCE	1302-2 High Resolution
RESOLUTION	580 x 470 Pixels
C.R.T.	14"
SUPPLY	220/240v. 50/60Hz.
E.H.T.	Minimum 19.5kv Maximum 22.5kv
VIDEO BAND WIDTH	10MHz.
DISPLAY	85 characters by 25 lines
SLOT PITCH	0.41mm
INPUT: VIDEO	R.G.B. Analogue/TTL Input
SYNC	Separate Sync on R.G.B. Positive or Negative
EXTERNAL CONTROLS	On/off switch and brightness control

To: Opus Supplies Ltd., 158 Camberwell Road, London SE5 0EE.
Please send me:

_____ High Resolution Colour Monitor(s) at £299.95 each (inc. VAT).

I enclose a cheque for £_____ Or please debit my credit card account with the amount of £_____ My Access/Barclaycard (please tick) no. is_____

Name_____

Address_____

_____

_____

Telephone_____

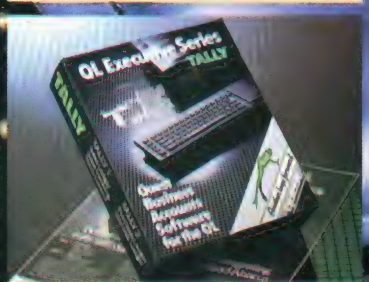
Opus.
Opus Supplies Ltd.

With Quest the QL really means business


Executive Series


QL Expansion Console


Memory Expansion Board



Business Accounts Software

CP/M 68K


Disk Drive

Quest's Executive series of quality peripherals and software for the Sinclair QL allows the business and serious home user to fully realise the full potential of the most exciting computer of the 80's.

Post the coupon below to obtain details of a wide range of Quest Executive products including:

- CP/M 68K
- Business Accounting Software
- Floppy and Winchester Disk Drives
- Memory Expansion Boards
- Monitors
- Printers
- Consumables

Telephone Sales Desk 04215 66488

Quest International Computers,
School Lane,
Chandlers Ford, Hants. SO5 3YY
Telex 47326.

Please send me details of the Quest Executive Series

Name _____ Tel. No. _____

Address _____ My QL Serial Number is _____

_____

Position _____

Recommended by QL-User magazine

At Last! THE SERIES THAT MAKES THE QUANTUM LEAP

THE SINCLAIR QL SERIES

"I am certain that these books will add enormously to the enjoyment and practical use which QL users will get from their computers."

NIGEL SEARLE

Managing Director, Sinclair Research Limited

5 books to help you get the most from the QL, whether you are still waiting or are lucky enough already to have yours.

Introducing the Sinclair QL explains how the QL works and what you can do with it.

Introduction to SuperBASIC on the QL explains SuperBASIC and introduces its special features and qualities enabling you to master programming quickly.

Advanced Programming with the Sinclair QL is an essential reference work for users who really want to get to grips with the Sinclair QL. The book includes such topics as program logic representation, types of commercial program, programming techniques and document design.

Desk-top Computing with the Sinclair QL shows just what can be achieved in business computing using the Sinclair QL and how to get the best out of the four QL software packages: word processing, spreadsheets, database management and business graphics.

Word Processing with the Sinclair QL has been written to explain both the concepts behind the uses of word processing and how the QL word processing package operates and what it can do.

General Editor, Robin Bradbeer and his team of authors received the help and co-operation from both Sinclair and Psion Software – creators of the QL software – to ensure that these books really are the ultimate handbooks for QL users.

Order now – you won't have to wait long for your books! Just fill in the order form and return it to us with your cheque or money order. You can even charge your Access or Barclaycard account. We will send you your books POST FREE within 28 days.

ORDER FORM To: TBS, 38 Hockerill Street, Bishop's Stortford, Hertfordshire. QL/1/85

Please send me:

..... copies of **Introducing the Sinclair QL** @ £6.95 each
..... copies of **Introduction to SuperBASIC on the QL** @ £6.95 each
..... copies of **Advanced Programming with the Sinclair QL** @ £6.95 each
..... copies of **Desk-top Computing with the Sinclair QL** @ £6.95 each
..... copies of **Word Processing with the Sinclair QL** @ £6.95 each

I enclose my cheque/money order made payable to TBS for £ _____

Please charge my Barclaycard/Access account (delete as appropriate) number _____

My name and address is _____

_____

_____

_____ Signed _____

Dealer enquiries to:
Doug Fox, Hutchinson, 17-21 Conway Street, London W1

HUTCHINSON

Write programs to unlock the multitasking power of your QL!

Use Metacomco's much acclaimed ***QL Development Kits*** *to create the software you need: Use* ***Assembler*** *for its speed, and for access to the QL's many features; use* ***BCPL*** *for systems programming; writing games, utilities, and applications packages; use* ***LISP*** *for manipulating data structures, and for exploring the world of artificial intelligence.*

RELEASE THE POWER OF THE QL

ASSEMBLER £39.95 INC. VAT + P&P

> Standard Motorola 68000 mnemonics.
> English error messages.
> Produces code which can be EXECed, and run as a concurrent job.
> External references allow linkage to high level languages.
> Integral linker allows assembler programs to be linked together.
> Macro expansions.
> Position independent, absolute or relocatable code can be produced.
> Conditional assembly.
> Large range of directives.
> Fully formatted listings.
> 30 character variable names and 32-bit values.

ENHANCED!
NOW INCLUDES ASSEMBLER LINKER*
AND NEW LOW PRICE £39.95

BCPL £59.95 INC. VAT + P&P

> True compiler.
> Multiple BCPL programs can run concurrently.
> Full runtime library includes interfaces to QDOS graphics, file operations, window handling, etc.
> Exception handling provides diagnostics for debugging.
> Link loader allows linkage of separately compiled segments.
> Assembler and BCPL modules can be linked together.
> 32-bit variables use the full QL address space.

LISP £59.95 INC. VAT + P&P

> Full support of QL features including window graphics, and screen handling.
> Compatible with Acornsoft Lisp for the BBC micro.
> Interpreter.
> Turtle graphics.
> Structure editor allows alteration of data structures.
> Prettyprinter displays programs in structured format.
> Tracer to aid in debugging.
> Garbage collector automatically recovers spare memory space.
> 28-bit integers and 250-character names.

Each language is supplied as a complete Development Kit which includes the software (on a Microdrive Cartridge), Metacomco's popular full screen editor, and user manual.

The QL Development Kits are available from HMV, Menzies and many other leading retailers, or direct from Metacomco.

METACOMCO
26 Portland Square, Bristol BS2 8RZ.
Tel: Bristol (0272) 428781

*Existing registered users can obtain an upgrade. Contact Metacomco for details.

QL & QDOS are the trade names of Sinclair Research Ltd.

Phone today, or post this coupon to: Metacomco, 26 Portland Square, Bristol BS2 8RZ.

NAME _____

ADDRESS _____

_____

POSTCODE _____

TEL. NO. _____

SIGNATURE _____

Please send me:
The QL Assembler Development Kit at £39.95 _____
The QL BCPL Development Kit at £59.95 _____
THE QL LISP Development Kit at £59.95 _____
I enclose a cheque for £ _____ or debit my ACCESS/VISA Account No.

Card expiry date: _____

Price includes VAT, postage and packing UK mainland only. Delivery allow up to 28 days. QL3

QL HORIZONS

The number of uses to which Psion packages may be put are limited only to the imagination and ingenuity of our readership. David Drysdale introduces a new feature series with a few simple proposals and then we transfer control to Archive

For me the QL has been a long time coming. Since 1980 I have waited for the appearance of an inexpensive electronic workhorse. During that time I watched in despair as games-minded boffins gave their machines explosive colour graphics, zappy sound effects and, finally, built-in 'sprites' – as if there was nothing more practical for computers to do! At last, however, my vigil is over.

Home Help

Household finance, I decided it would be the first project for my QL. Some years ago I bought a Personal Budgeting System, from the business stationery company Kalamazoo. It was excellent but eventually discontinued. Based on a printed spreadsheet it seemed to be ideal for Abacus and I even had visions of developing a marketable Kalamazoo System program which the company might be interested in promoting – but Abacus changed all that.

Not that the program had any difficulty in handling the spreadsheet – it was just too good. I should have realised that a different approach was needed for a printed spreadsheet as opposed to a computerised one. All the painstaking care that Kalamazoo had put into designing their paper spreadsheet was not needed in a computerised self-calculating system. I had a neat little budget system up and running in no time but it bore no resemblance to the one designed by Kalamazoo.

Penmanship

Next job on the QL was to write this article using the word-processor Quill. *QL USER* had warned this was an unwise thing to do as the system was not yet reliable and I soon understood why! As I tried to sort out the microdrives, the message 'fatal error in edit mode' flashed on the screen and my first few precious paragraphs disappeared for ever.

From that moment on frequent backing up became the rule. Even so, there was a sinister bug lurking around, for the machine behaved faultlessly whenever I had my data backed up but jammed instantly the moment I relaxed and decided not to bother.

From one viewpoint the whole exercise of writing on the computer was pointless as I don't have a printer yet but it was the writing process itself that interested me. As a slow typist I have always found the constant rejigs and rewrites take a lot of time. So I reasoned that, even without a printer, correcting as I went along using Quill, then typing the finished result directly from the screen would have some advantges – and so it turned out.

Anyone with a word-processor such as Quill can quickly learn to build up an article a paragraph at a time, changing the order of paragraphs where necessary and giving each one a final polish. Even the simple process of deleting hackneyed phrases can bring a ordinary piece of writing up to saleable standards.

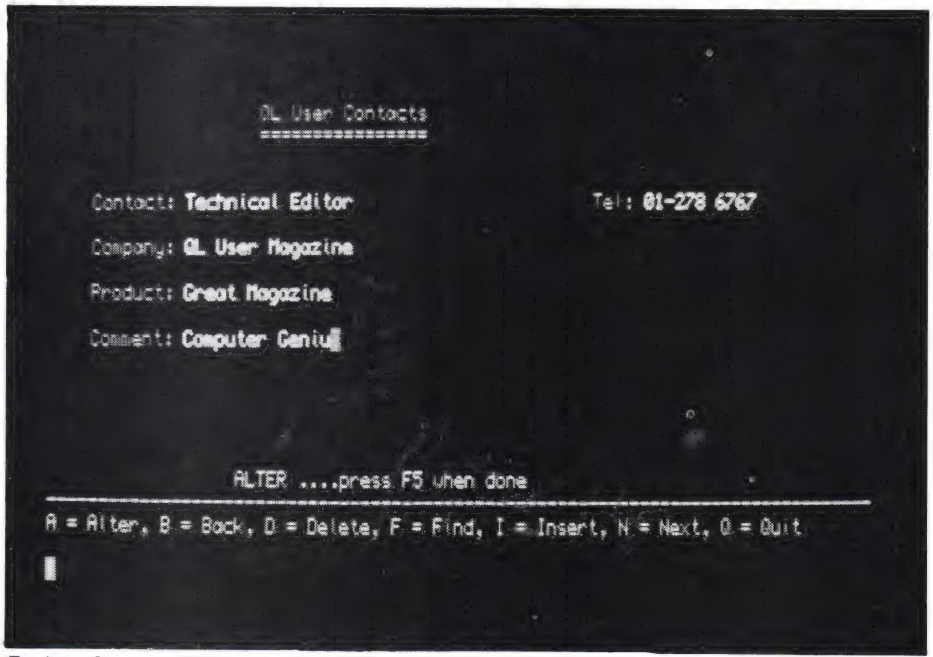
Listing Heavily

After my experiments with Abacus and Quill I turned to Archive – but not for long. Trying to make a back-up copy the message "bad or changed medium" came on the screen and I returned the cartridge promptly – a pity really as I had great plans for Archive. Apart from the usual filing applications, in my case the indexing of a cuttings file, I want to build up a collection of lists.

Most people write lists occasionally, if only for shopping, and the big advantage of a computerised list is that it can be altered so easily. One writer, Nina Grunfeld, has made a special study of lists and finds them to be a powerful organisational aid in the home.

In her book "The Complete Book of Household Lists" (Judy Piatkus, Pan £1.95, 1984) she has compiled over 200 useful lists "dedicated to those who like organising, or to be organised". She covers money, health and family under headings such as: "Your Filing System", "Your Important Numbers" and "Better Safe than Sorry".

Nina's book was based on written lists but there is no difficulty at all in using the ideas for building up a useful database.



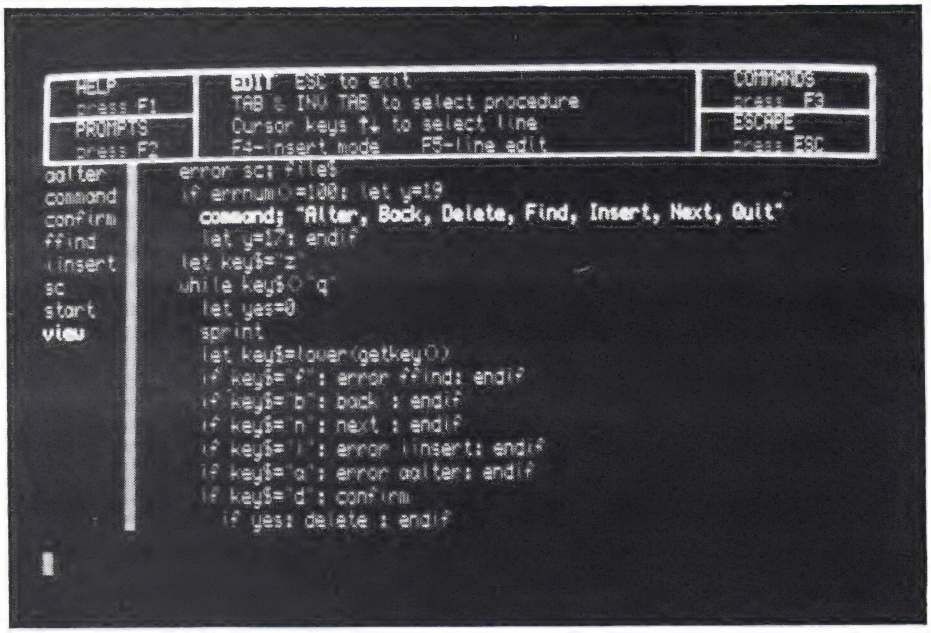
Design a layout and the rest comes easy

Closing Prices

The final part of the Psion's package, Easel, I have not yet tried but have ideas for.

Locked away in a cupboard somewhere I have an old insurance policy based on unit trusts. It is a switchable policy so that, at any time, I can switch the units into different sectors of the Market such as Property or Shares and so take advantage of changing trends.

A good idea, but in practice it's too much hassle to keep abreast of Stock Market trends just for the sake of one small insurance policy, and this is where Easel might help. To keep a visual track of the units by displaying them on an Easel graph and updating it weekly should be an easy thing to do and give clear indications of when it's time to switch. Who knows, it may ►



Archive is unique with its full screen editor

even make enough money to repay what has been spent on my QL computer!

So my little workhorse is really set to work. Critics who sneer at the QL have really missed the point. As it stands, it is a complete sit-down-and-go package that only needs the addition of a printer to complete it. Can anybody, for a mere £400, expect anything better than that?

Of all Psion's packages Archive is the least accessible in that it requires the user to have programming skills as well as participating in the design of a database.

In this article we concentrate upon the programming side and develop a simple set of general routines to be used with **any** single database that the user may wish to set up, provided the following two rules are observed:

A> That screen layouts carry identical names to databases to which they are related.

B> That row 17 on the main display be left blank.

For inexperienced users we have included a simple telephone directory illustrating how to set up a database and accompanying screen layout. Those who have already accomplished this should move directly to the procedures themselves.

Preliminary Exercise

It's a good idea to design your screen layout prior to creating the database proper. Whilst ultimately the layout will provide an invaluable guide for those unfamiliar with the database or routines associated with it, at this stage its design encourages one to examine the number, order and type of fields that will be required. Most importantly, altering the layout is considerably easier than amending the database itself.

To start the ball rolling press **FS** and enter **sedit.** After a few moments the cursor will position itself at the top left hand corner of the hopefully blank display. Then it may be moved using the cursor keys.

The first stage in design is to enter background information on the screen such as database description, labels and prompts. So as to distinguish fixed information from data, change the ink colour to say, green. Press **F3** and then **I** repeatedly to make your selection and then **ENTER.** Following this type in the appropriate text at the desired locations.

In the case of the general routines covered here we would recommend that the bottom line of the display contain the following prompts:

A = Alter, B = Back, D = Delete, I = Insert, F = Find, N = Next, Q = Quit

By now you should have a clear picture as to the number and type of fields required for your database. The next step then is to incorporate them on the layout. The order in which you do so will determine the order that they will be entered so, having changed ink colour again (select white), move the cursor to just beyond the first label and press **F3** followed by **V.** Enter the field name followed by a **$** (dollar sign if it is to contain letters, punctuation or spaces). It is also advisable to jot its name down for future reference when creating the database proper. Finally press **ENTER.**

You will then be required to specify the field's maximum length by repeatedly pressing ".". Note, that this is purely for presentation purposes as all fields on Archive are of variable length. All data entered will be stored. However, when moving onto another record certain areas of the screen may not be wiped clean if this procedure is ignored. When complete press **ENTER** and then move onto entering the next field in a similar fashion.

Once all fields have been entered (and noted) press **ESC** and **ssave** the screen layout under the name you plan to call your database, for example, **ssave "Tel".** Should you wish to amend the layout at a later date, simply **sload** and re-enter **sedit.** When complete **ESC** and **ssave** once more.

Setting Up Shop

Creating the database itself is considerably easier than setting up the screen layout, especially if a list of all fields has been compiled from the previous operation. Simply enter **create,** then type in the database's name, in this case **"Tel"** and press **ENTER.** After this, continue by entering each field from your list. In the example cited, this would be the following:

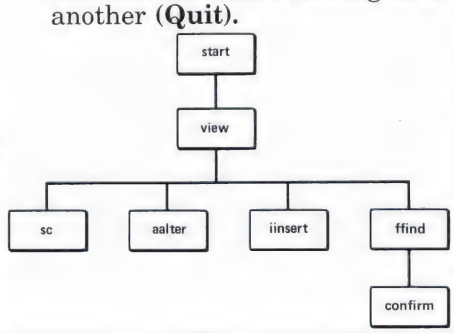
Name$ [ENTER]
Phone$ [ENTER]
Company$ [ENTER]
Products$ [ENTER]
Comment$ [ENTER]
[ENTER]

With the final **ENTER** the process will be complete and the database automatically written back to microdrive2. If any errors have occured in its creation, for example a field omitted, it will be necessary to delete it (ie **kill "tel_dbf"**) and set it up all over again.

All-Purpose Toolkit

As with all Archive programs the routines consist of a set of procedures linked together in a single program and hierarchically organised in an inverted tree structure. (See disgram). Briefly the routines allow one to select a database and then:

1 – Step forwards **(Next)** or **(Back)** backwards through it.
2 – Locate a series of records in which a particular string occurs **(Find).**
3 – **Insert** a new record.
4 – **Amend** or **Delete** an existing record.
5 – Save the amended database to microdrive2 before moving onto another **(Quit).**



To key in the procedures you will have to enter Archive's program editor. To do this type **edit** and then enter the first procedure so that what is displayed is identical to the listing: For example

proc **aalter**
command;"ALTER . . . press F5 when done"
alter
endproc

To correct any errors on entry use the **F4** and **F5** options. When a procedure has been entered, move onto the next by pressing **SHIFT** with **F3.**

The following short summary of the procedures and what they do may prove useful:

Start
The procedure contains most of the error trapping routines. This is achieved by preceding the first occurence of a lower call with **error.** This idiot-proofs the program and provides the user with a number of opportunities to set things to right without crashing.

View
This is the workhorse of the system. It opens and closes the designated database, checks to see which key is pressed and then either calls a subsidiary procedure **(ffind, aalter,iinsert)** or performs the desired opertions itself (Delete, Back, Next).

Aalter, ffind, iinsert,sc
These are subsidiary procedures, each of which performs a single operation. They have been separated out so that errors in the execution will not cause the program to crash but return control to **view.**

Command, confirm
These are general purpose routines used throughout. The former centres a prompt on line 17 of the screen display. The latter as its name suggests requests that the user confirm a particular course of action.

Once the procedures have all been entered leave the editor by pressing **ESC** and save them on microdrive under the name "view" (ie **save "view").** Thereafter the procedure may be loaded and run simply by typing:

run "view". (If view is already loaded enter **start.**

The routines have been designed so that at every step instructions or prompts are provided, consequently few problems will be encountered even by the least experienced. However, given the QL's penchant for crashing after prolonged usage, users would be well advised to backup any database that has been modified at the end of every session.

```
proc aalter
  command; "ALTER ....press F5 when done"
  alter
  endproc
proc command; m$
  print at y,0; rept(" ",64)
  if len(m$)<64: let x=32-int(len(m$)/2)
    if m$<>"": print at y,x; : print ink 5; m$; : endif
    endif
  endproc
proc confirm
  command; "Confirm (y/n)"
  let yes=lower(getkey())="y"
  endproc
proc ffind
  command; "Look for? ": input what$
  find what$
  while found() and not yes
    sprint
    error confirm
    if not yes: continue : endif
    endwhile
  if not found(): command; "No record": what$: endif
  endproc
proc iinsert
  command; "INSERT MODE...F5= Save, ESC = End"
  insert
  endproc
proc sc; sc$
  sload sc$
  endproc
proc start
  local err
  let err=1: let y=17: let x=0
  while err
    error command; "QL User Utilities": error view
    let err=errnum()
    if err=12: command; "Repeat request": close : endif
    if err=93: let y=15: command; "Press F2 and then ENTER"
      input y$: cls : let y=17: endif
    if err=100: command; "No such file": endif
    endwhile
  endproc
proc view
  local file$,key$
  cls
  command; "Enter Filename or 'END': ": input file$
  if lower(file$)<>"end"
    open file$
    error sc; file$
    if errnum()=100: let y=19
      command; "Alter, Back, Delete, Find, Insert, Next, Quit"
      let y=17: endif
    let key$="z"
    while key$<>"q"
      let yes=0
      sprint
      let key$=lower(getkey())
      if key$="f": error ffind: endif
      if key$="b": back : endif
      if key$="n": next : endif
      if key$="i": error iinsert: endif
      if key$="a": error aalter: endif
      if key$="d": confirm
        if yes: delete : endif
        endif
      command; ""
      endwhile
    close
    endif
  endproc
```

MONITOR/PRINTER PACKAGES

FOR

SINCLAIR QL

- **NO INTERFACING PROBLEMS. JUST PLUG IN AND GO!**
- **ALL SYSTEMS FACTORY-TESTED BEFORE DESPATCH TO YOU!**

PACKAGE DEALS

MONITOR	*with Printer*	*Package Price*
	Brother HR5 + mains adaptor	£425
Microvitec 14" colour 1451DQ3	Epson RX80 F/T	£585
	Epson FX80	£725
	Epson FX100	£869
	Epson DX100	£745
	Brother HR15	£715
	Canon PW1080A NLQ	£649
	Epson JX80 colour	£895
	Canon PJ1080A colour	£835

(Illustrated: QL with Microvitec 1451DQ3 and Epson RX80F/T)

Epson parallel printers supplied with RS232C I/F fitted Canon printers supplied with Miracle Systems I/F. All monitor and printer leads included.
Prices are inclusive of VAT. Add £12 for delivery.

PRINTERS		*ACCESSORIES*		*MONITORS*	
Dot matrix:		*Tractor feeders:*		*Colour:*	
Epson RX80	£260	FX80	£35	Microvitec	
RX80F/T	£290	LQ1500	£60	1431 Std. res.	£199
FX80	£425	DX100	£90	1431 MZ Spectrum	£235
FX100	£620	HR1	£95	1451 Med. res.	£299
LQ1500	£1,125	HR15	£90	1451 DQ. QL	£275
JX80 colour	£625	HR25/35	£99	1441 High res.	£506
Canon PW1080A NLQ	£375	*Cut-sheet feeders:*		1431 PAL/Audio	£259
PJ1080A colour	£529	DX100	£250	1451 PAL/Audio	£375
Daisy Wheel:		HR15	£250	*Monochrome:*	
Epson DX100	£495	HR25/35	£250	Philips	
Brother HR1	£675	*Keyboards:*		BM7502 green	£92
HR15	£450	DX100	£170	BM7522 amber	£92
HR25	£795	HR15	£170	Sanyo	
HR35	£995	RS232C I/F	£30	DM2112	£90
		RS232C 2K buffer	£65	DM8112	£125
				Novex 12/800	£120

Prices are inclusive of VAT

For Printers/Monitors add £10 for express delivery and for Accessories add £5 for express delivery

ZEAL MARKETING LIMITED

VANGUARD TRADING ESTATE, STORFORTH LANE, CHESTERFIELD S40 2TZ
Tel: 0246 208555 Telex: 547697

Cursing Recursion!

The QL's version of BASIC is one of a limited number capable of supporting recursion. Dr Andy Carmichael examines the concept and unlocks a Pandora's Box of elegant and powerful routines.

The QL comes with a free text adventure euphemistically called the "User Guide". Buried deep within the "Keywords" section is the short statement:

"Procedures (and function) calls can be recursive . . ."

Seemingly innocuous, this introduces one of the most elegant and powerful tools available to the programmer.

A recursive procedure is one which directly or indirectly calls itself. The following electronic book-at-bedtime story illustrates its use.

```
110 DEFine PROCedure Tell_A_Story
120 PRINT "'It was a dark and stormy night, and the mate turned to the captain";
130 IF RND>5E-2 THEN
140 PRINT "and said, 'Captain tell us a story'."
150 PRINT "And this is the story he told."
160 Tell_A_Story
170 ELSE
180 PRINT ". At that moment the ship sank.";
190 END IF
200 PRINT "And that's the end of the story.'"
210 END DEFine Tell_A_Story
```

Notice that at line 160 the procedure calls itself. The danger with recursive procedures is that they may continue calling themselves forever (or in fact until the machine runs out of memory). So there should always be a terminating condition (in this case when the random number given by RND is less than 0.05) at which point the procedure calls begin to unwrap themselves to the outermost level.

Treadmill Called

A procedure which calls itself via a sequence of other calls is said to be *mutually recursive* – such as the following entries reputed to come from a Russian dictionary:

HUMOUR: see irony.
IRONY: see humour.

Translated into Super Basic this would appear thus:

```
300 DEFine PROCedure Humour
310 PRINT "Humour = ";
320 Irony
330 END DEFine Humour
340 DEFine PROCedure Irony
350 PRINT "Irony = ";
360 Humour
370 END DEFine Irony
```

Hardly a very useful program, particularly as it demonstrates this problem of a missing terminating condition, so it can never finish. It's like a loop construct with no exit condition – but unlike an endless loop the recursive program will not run forever. If you enter this program and type "Humour" you find that after some time the execution of the program begins to slow down as the memory fills up.

Eventually the QL comes back with the error message "Out of memory", since for each procedure call any local variables must be stored. The computer must store its "current state" so that if, and when, control does return from a procedure, it knows whereabouts in the calling procedure it is and the values of all the local variables.

Archetypal Formula

So far we've not actually done anything useful with recursion so let's try a classic example – the FACTORIAL function. As all you Einsteins out there know, **n!** (pronounced "n factorial"), where n is a positive integer, is obtained by multiplying all the integers between 1 and n together. Which is fine as a definition unless n happens to be zero since, rather surprisingly perhaps 0! = 1. What about a more concise and accurate definition . . .

n! = n * (n–1)! for n >0
n! = 1 for n = 0

Behold, a recursive definition of the problem! And note that this time we have a terminating condition (when n = 0) so we shouldn't go on and on forever. Following the best principles of "object-orientated design" (that's the latest buzz-word for all those up in "software engineering") our program should be as close as possible to the definition of the problem – so why not a recursive solution? We'll come to some of the reasons why not in a moment! . . . but first the program:

```
500 DEFine FuNction factorial(n)
510   IF n==0 THEN
520     RETurn 1
530   ELSE
540     RETurn n*factorial(n-1)
550   END IF
560 END DEFine factorial
```

Having entered the program, type

ILLUSTRATION BY DAVID HINE

"PRINT FACTORIAL(10)" for example to test it. Note the recursive function call at line 540.

Well – don't you think it's a neat solution to the problem? Your answer will really depend on how easily you think in terms of recursion, which for most of us is not the most natural way to analyse problems. Probably a much simpler way to solve this problem is by using a FOR loop construct such as in the following example:

```
500 DEFine FuNction factorial(n)
510     value = 1
520     FOR factor = 1 to n
530         value = value*factor
540     END FOR factor
550     RETurn value
560 END DEFine factorial
```

This program also avoids the memory overflow problems so it can handle slightly larger numbers than the first, and for large numbers it is also faster. The second program would also be preferred according to the all important KISS rule of programming ("Keep It Simple Stupid"). So the factorial example, although a good introduction to the ideas of recursion is not an ideal candidate for implementing as a recursive function in a real applications program.

Brain Teaser

However there are many cases in which recursion is both the natural way to solve a problem and the most efficient. Generally recursive procedures need more careful thought to write and understand but once grasped they can be elegant, concise and, most importantly, simple! The "Towers of Hanoi" is a good example of a problem which is naturally recursive and it's the next subject we consider. Later we shall look at a recursive sort algorithm "Quicksort" which turns out to be one of the most efficient methods available for sorting very large arrays.

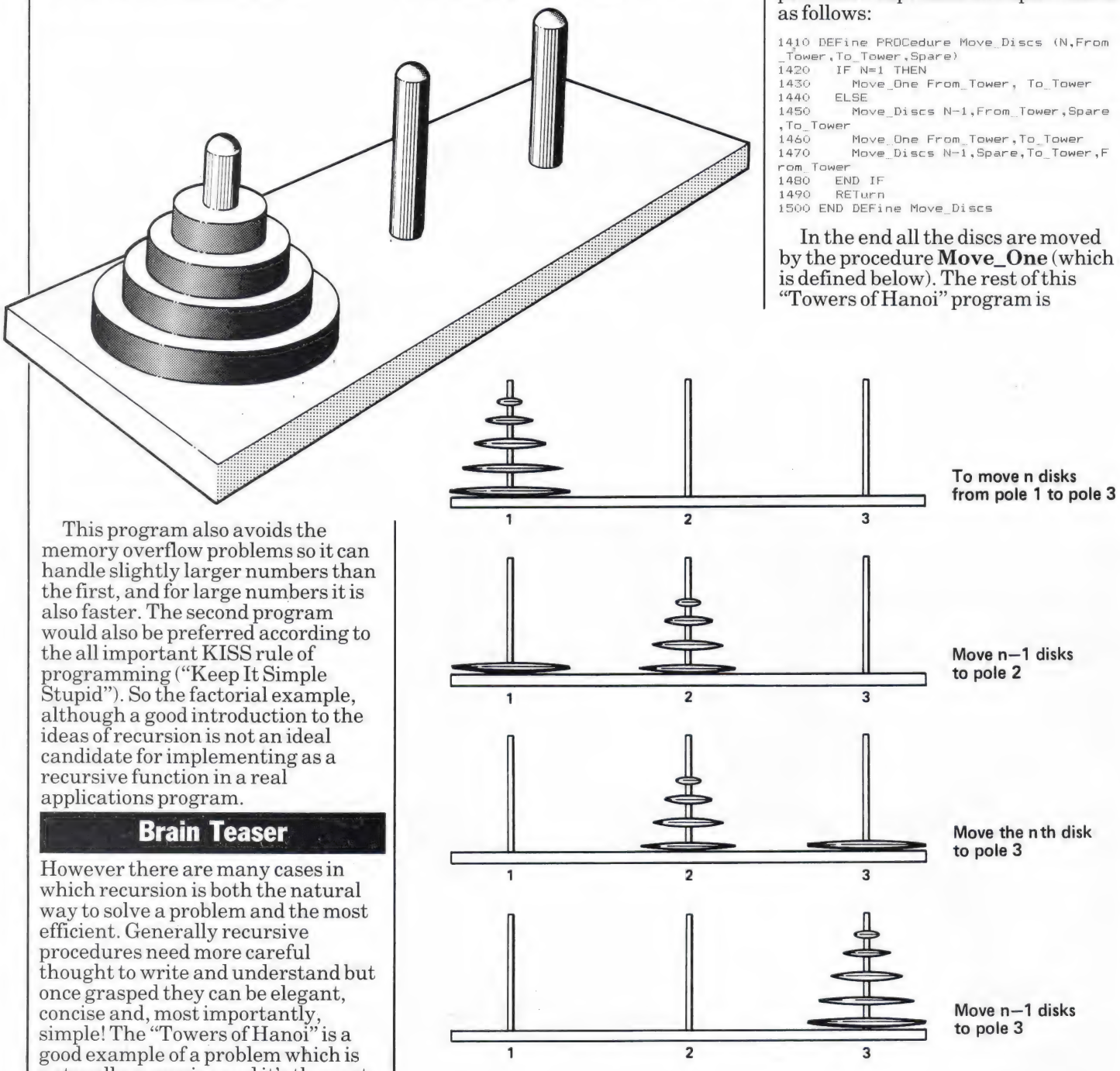
Maybe you have seen a child's toy consisting of three poles with different sized wooden discs that fit over the poles. If so you know what the "Towers of Hanoi" look like. First let's write a program that explains the problem.

```
1660 DEFine PROCedure Explain_the_proble
m
1670    INK#5, 4: CSIZE#5, 3,1
1680    PRINT#5, "       TOWERS of HANOI"\
1690    CSIZE#5, 2,0: INK#5, 7
1700    PRINT#5, "The towers of Hanoi is
an ancient  mathematical problem. It con
sists  of three poles, on one of which i
s placed a number of different sized dis
cs in ascending order of size."
1710    PRINT#5\ "The problem is to move
the discs   from the first pole to the l
ast onewithout ever placing a larger dis
c on top of a smaller one."
1720    PRINT#5\ "This computer program s
hows how    this may be done and counts
how    many moves are required to do it.
"\\
1730 END DEFine Explain_the_problem
```

This procedure prints out an explanation of the problem, so ignore the rest of it for now and just read the text in the print statements. The solution is most easily arrived at by using recursion. If we want to move **n** discs from pole 1 to pole 3 (see figure 1), firstly we move **n-1** discs to pole 2, using pole 3 as the spare (line 1450 of the program below), then we move the biggest disc to pole 3 (line 1460). Finally we move the **n-1** discs to pole 3, using pole 1 as the spare (line 1470). Of course if **n** happens to equal 1 then we just move it with no further complication (line 1430). The procedure expressed in SuperBasic is as follows:

```
1410 DEFine PROCedure Move_Discs (N,From
_Tower,To_Tower,Spare)
1420    IF N=1 THEN
1430       Move_One From_Tower, To_Tower
1440    ELSE
1450       Move_Discs N-1,From_Tower,Spare
,To_Tower
1460       Move_One From_Tower,To_Tower
1470       Move_Discs N-1,Spare,To_Tower,F
rom_Tower
1480    END IF
1490    RETurn
1500 END DEFine Move_Discs
```

In the end all the discs are moved by the procedure **Move_One** (which is defined below). The rest of this "Towers of Hanoi" program is



The discs problem – transferring pile 1 to pole 3.

concerned only with displaying the movement of the discs on the screen. It doesn't involve any further recursion so is presented here without any further comment. If you do type it in and run it you'll find the movements of the discs are displayed graphically showing just how many individual movements are required to move a small number of discs (add just one more disc and you double the number of moves required).

Hi-Tech Sort

```
1000 HANOI
1010 DEFine PROCedure HANOI
1020   MODE 8: OPEN#5, con_500x200a0x0_0
1030   BORDER#5, 20,3: PAPER#5, 0: CLS#5
1040   Explain_the_problem
1050   REPeat towers
1060     INPUT#5, "Number of discs?",Num
ber
1070     CLS#5
1080     IF Number<1: EXIT towers
1090     DIM height(3),Sizes(3,Number),x
(3)
1100     m=0
1110     height(1)=Number
1120     FOR i=1 TO 3
1130       x(i)=(2*i-1)*(Number+1)/2
1140     END FOR i
1150     FOR i=1 TO Number
1160       Sizes(1,i)=Number+1-i
1170     END FOR i
1180     Draw_Discs Number
1190     Move_Discs Number,1,3,2
1200     INK#5, 7
1210     PRINT#5, "I moved the ";Number;
" discs in ";m;" moves"
1220   END REPeat towers
1230 END DEFine HANOI
1240 DEFine PROCedure Draw_Discs(Num)
1250   SCALE#5, 3*Num/2+2,-.5,-1
1260   INK#5, 7
1270   LINE#5, 0,0 TO 3*Num+3,0
1280   FOR i=1 TO 3
1290     LINE#5, x(i),0 TO x(i),Num+.5
1300   END FOR i
1310   FOR j=Num TO 1 STEP -1
1320     Draw_One 1,Num-j+1,j,(j MOD 7)+
1
1330   END FOR j
1340 END DEFine Draw_Discs
1350 DEFine PROCedure Draw_One(Column,Ro
w,Size,colour)
1360   INK#5, colour: FILL#5, 1
1370   POINT#5, x(Column)-Size/2,Row
1380   LINE_R#5 TO Size,0 TO 0,-.8 TO -S
ize,0 TO 0,.8
1390   FILL#5, 0
1400 END DEFine Draw_One

1510 DEFine PROCedure Move_One(From_,To_
)
1520   LOCal h,s
1530   h=height(From_)
1540   s=Sizes(From_,h)
1550   height(From_)=h-1
1560   Draw_One From_,h,s,0
1570   INK#5,7
1580   LINE#5, x(From_),h TO x(From_),h-
1
1590   h=height(To_)+1
1600   height(To_)=h
1610   Sizes(To_,h)=s
1620   Draw_One To_,h,s,(s MOD 7)+1
1630   m=m+1
1640   RETurn
1650 END DEFine Move_One
```

The final example of recursion to be considered in this article is a program for sorting an array of real numbers into ascending order. The problem of sorting a large number of items is so common in computer programs, and so time consuming, that it has been the subject of a great deal of research in order to discover the most efficient way of doing it for a particular type of data. This particular method was first published by C.A.R. Hoare in The Computer Journal (Vol 5, 1962) and is called "Quicksort". It has been shown to be considerably more efficient for large amounts of data language – the resulting program is still bound to be pretty slow, but at least we know that it uses one of the best sorting methods around!

The Quicksort procedure works by partitioning the array (called **A** say) into two parts – the first part containing values which are all less than or equal to some arbitrary value picked from the array, the second part containing values which are all greater than or equal to that value. Having partitioned the array, the sort procedure may be called recursively to do the same again to the two parts of the array. This carries on until the parts are reduced to a size of either one (in which case no further sorting is required) or two (in which case a simple swap may be necessary). At the end of this process the entire array is sorted. Let's call the procedure to sort our array **Sort_A** – it looks like this:

```
1100 DEFine PROCedure Sort_A(l,u)
1110   LOCal n,I,J
1120   n=u-l+1
1130   SELect ON n
1140     ON n=1
1150       RETurn
1160     ON n=2
1170       IF A(l)>A(u) THEN
1180         Swap l,u
1190       END IF
1200       RETurn
1210     ON n=REMAINDER
1220       Partition_A
1230       Sort_A l,J
1240       Sort_A i,u
1250   END SELect
1260   RETurn
1270 END DEFine Sort_A
```

The Array **A** must already have been defined before this procedure is called and the two procedures it calls (apart from itself) **Swap** and **Partition_A** are defined below. The procedure has two parameters, **L** and **U**, which are the lower and upper bounds of the part of the array which we wish to sort (initially **L** = 1 and **U** is equal to the size of the array). The local variable **n** is set to the number of elements in this slice of the array and there are three **"SEL**ect" options depending on whether **n** is 1,2 or more than 2. When there are more than 2 elements we partition the array and call **Sort_A** recursively to sort both parts. The variables **I** and **J** are assigned by **Partition_A,** which is as follows:

```
1280 DEFine PROCedure Partition_A
1290   LOCal R
1300   R=A(INT((u+l)/2))
1310   I=l : J=u
1320   REPeat Swapping
1330     REPeat Left_Part
1340       IF A(I)>=R : EXIT Left_Part
1350       I=I+1
1360     END REPeat Left_Part
1370     REPeat Right_Part
1380       IF A(J)<=R : EXIT Right_Part
1390       J=J-1
1400     END REPeat Right_Part
1410     IF I<=J THEN
1420       Swap I,J
1430       I=I+1
1440       J=J-1
1450     END IF
1460     IF I>J : EXIT Swapping
1470   END REPeat Swapping
1480 END DEFine Partition_A
```

In the above procedure **I** and **J** move through the array from either end comparing each element with the value **R.** When both have found an element which is either in the wrong half or is equal to **R** (and this must happen if **R** is a value chosen from the array in the first place) then the contents of **A(I)** and **A(J)** are swapped and the process repeated until **I** and **J** meet at some point in the middle.

Here's the code for the procedure **"Swap":**

```
1490 DEFine PROCedure Swap(P,Q)
1500   LOCal Temp
1510   Temp=A(P)
1520   A(P)=A(Q)
1530   A(Q)=Temp
1540   RETurn
1550 END DEFine Swap
```

Finally we need some code to actually test our program to see if it works, before trying to incorporate it in some bigger program where it would mess things up if it didn't work. This simple program reads numbers from the keyboard, prints them out, sorts them, and finally prints them out in order.

```
1000 DEFine PROCedure Main
1005   PRINT \"QUICKSORT"\
1010   INPUT "How many numbers to sort?
";Max%
1015   INPUT "Numbers to be generated (1
) or typed (0)? (Answer 1 or 0) ";Random
1020   DIM A(Max%)
1025   FOR k=1 TO Max%
1030     IF Random THEN
1035       A(k)=RND(100)
1040     ELSE
1045       INPUT "Next number : "; A(k)
1050     END IF
1055   END FOR k
1060   FOR k=1 TO Max%: PRINT A(k),;
1065   Sort_A 1,Max%
1070   PRINT \"Sorted numbers:"
1075   FOR k=1 TO Max%: PRINT A(k),;
1080 END DEFine Main
```

Recursion should be seen as another tool for writers of programs, which can be very powerful in creating compact, efficient and comprehensible code. Unfortunately it can be misused and if you're not careful you can end up creating programs that are very difficult to debug because no one, not even you, can understand what's going on!

Parting Shot

Hopefully the programs in this article have shown that recursion can be simple and powerful, as well as fun to use! Sometimes you come across a problem that would be much more difficult to solve any other way – try rewriting the "Towers of Hanoi" procedure using a FOR loop for example. These are the problems to use recursion on. If it's almost as easy to solve with a loop as with recursion then usually it's safest to stick with the familiar old loops.

One final point – most of the modern languages such as Pascal, ALGOL68, C, Ada, LISP, APL and so on allow recursion. The fact that SuperBasic does too is one more reason why it's by far the best BASIC presently available on home micros for teaching programming to the next generation.

Prism Monitor

Technical Editor Paolo Baccanello takes a penetrating look at Prism's colour monitor, and tells us whether it improves the QL's image.

At first glance the QL14 comes across as a well finished elegant complement to the QL itself. Matt black, the module is guaranteed not to be an eyesore for those who consider a good colour scheme more important than technical merit. Certainly, Prism are to be congratulated for not resorting to the fashionable euphemism in describing their monitor as "colour compatible" with the QL, which in layman's terms means, quite simply that it's black. However, whether a monitor can be said to have a "livery" . . . well, that's beside the point.

When switched on it soon becomes apparent that the QL14 is in a class of its own, occupying a slot in between a television display and a fully fledged monitor. In keeping with the manufacturer's claims the device is capable of supporting the QL's singular 85 character display and as a consequence the shimmer and blur associated with TV sets has been reduced to a minimum. However, the 12MHz bandwidth is insufficient to provide a rock steady display and the screen tends to fluctuate up and down. Whilst barely noticeable at first, this effect begins to take its toll in terms of fatigue and eye strain after prolonged use.

On all remaining counts the device compares reasonably with most medium resolution monitors. Characters are easily legible at normal reading distance, colours adjacent in the spectrum do stand out against each other and no distortion occurs at the edge of the screen. On the final point, however, the display did not occupy the entirety of the screen but fell short by a few pixels at the top revealing a magenta and blue border quite alien to Quill. Whether this could be remedied by fine tuning on the part of the retailer or is a permanent feature of the machine remains to be seen.

The QL14 comes with a detachable anti-glare tinted glass front which is simple to install or remove. Without the screen, colour clarity is considerably reduced and the original deep black of Psion's backgrounds turns to an off-grey similar to that found on TV sets.

Controls are located at the front of the device with an on/off switch recessed at the bottom and a contrast control hidden in a compartment behind Prism's logo. The QL14 plugs directly into the RGB port and a connecting cable is included. Finally, a built-in retractable carrying handle is located on top and provided you are a weight-lifter, the monitor justifies the designation of "portable".

All in all then, resolutions and picture quality on the QL14 fall short of that obtainable on its up-market cousins whether 14″ or 12″, but then so does its price. Certainly it is an improvement on television sets, though even here, recent innovations from Japan permitting direct input via RGB/Composite video are beginning to make serious inroads at the bottom-end market where dedicated monitors such as this reside.

QL14 Colour Monitor

Price: £200
Supplier: Prism Microproducts, Prism House, 18-29 Mora Street, City Road, London EC1 8BT
Size: Height 375mm
Width 375mm
Depth 390mm
Bandwidth: 12 MHz
Dot resolution: 625 (H)
Weight: 12 Kg

CASE SOLVED

£69.00
+P&P.

- ***antler*** manufactured specially for the QL System.
- *Angled Keyboard for practical in-case use.
- *Space for 24 Cartridges, QL-accessories & Manual.
- *Size conforms to Airline hand luggage specifications.
- *Fully Guaranteed for 2 years.

Offer is for the case only.
'QL' is a registered trademark of Sinclair Research Ltd.

To: Management Science Ltd. QL Case offer. 17 West Hill, London. SW18 1RB.
Please supply ..... Case (s) for my QL(s) at £69 & £4 p&p.(inc. VAT)
I enclose a cheque/postal order for £.....
Made payable to: Management Science Ltd.

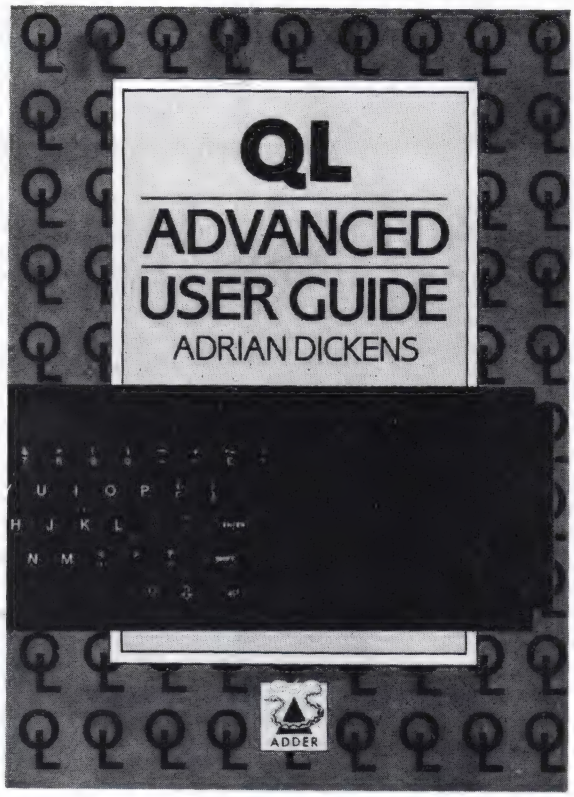
Name: .....
Street: .....
Town: .....
County: .....
Postcode: .....

Please allow 14 days for delivery.
QL2/85

THE ASSEMBLER

The ADDER 68K Assembler is a professional, fully integrated editor/MACRO assembler development package for the QL. Providing a quick turn-around on the *editing, assembling, debugging* cycle, it assembles Motorola format source files to produce multi-tasking programs, additional Super BASIC facilities plus much more. Features include:

- ★ FULL 68000 MACRO Assembler
- ★ Integrated full screen editor
- ★ Conditional assembly
- ★ External library file inclusion
- ★ Complete error checking diagnostics
- ★ Comprehensive manual
- ★ MACRO ASSEMBLER only £29.95*



THE BOOK

The QL Advanced User Guide (£12.95*) has been written by Adrian Dickens in collaboration with Tony Tebby (QDOS System designer). It is the complete guide to QDOS and the Sinclair QL, covering multi-tasking, transient programs, resident procedures, heaps and stacks, traps and utilities, 68008 assembler programming plus much more. All of these features are illustrated by practical examples, and the powerful QDOS experimentor program allows many facilities to be tried out from BASIC. All of the programs from the book are available on a microdrive cartridge which can be purchased with the book (£9.95*).

ORDER FORM

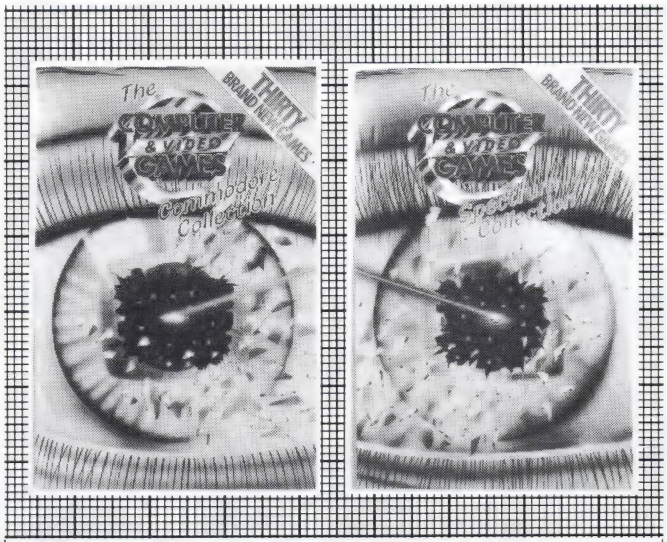
☐ Assembler development package **£34.95** incl. VAT and p&p
☐ QL Advanced User Guide Book **£14.45** incl. p&p (no VAT)
☐ QL A.U.G. programs from book **£11.95** incl. VAT and p&p
☐ **SPECIAL OFFER**
All three items from above (normally £61.35) **£55.95** ALL inclusive

I enclose cheque/PO for £.....
Name .....
Address.....
.....
City..... Postcode .....
Send to: **ADDER PUBLISHING LIMITED, P.O. Box 148, Cambridge CB1 2EQ**
Dealer enquiries welcome

QL and QDOS are trademarks of Sinclair Research Limited
** Price excluding VAT and p&p*

ADDER
QL/2/85

Commodore Collection — COMPUTER & VIDEO GAMES — Spectrum Collection



Computer and Video Games, Britain's most popular computer games monthly, bring you two exciting new books for the Spectrum and Commodore 64.

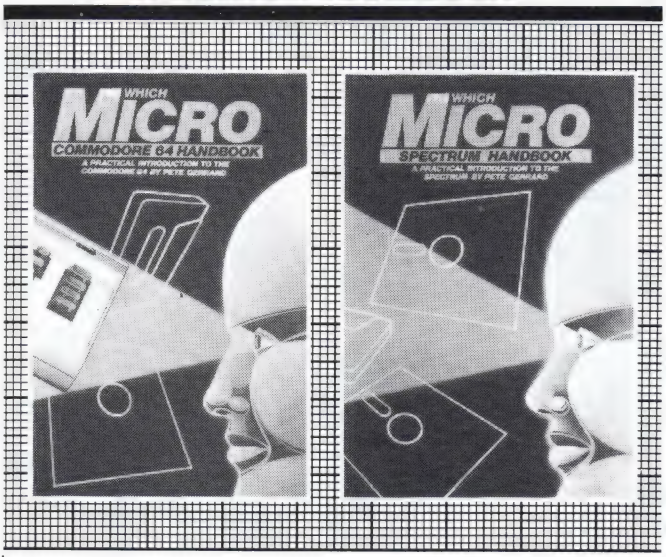
Each 100-page book contains up to 30 **brand new** games checked and prepared by **Computer and Video Game's** regular contributors.

At £4.99 - the best value on the market at all good bookshops from November or send to EMAP Books, Bushfield House, Orton Centre, Peterborough PE2 0UW.

WHICH MICRO

COMMODORE 64 HANDBOOK

SPECTRUM HANDBOOK



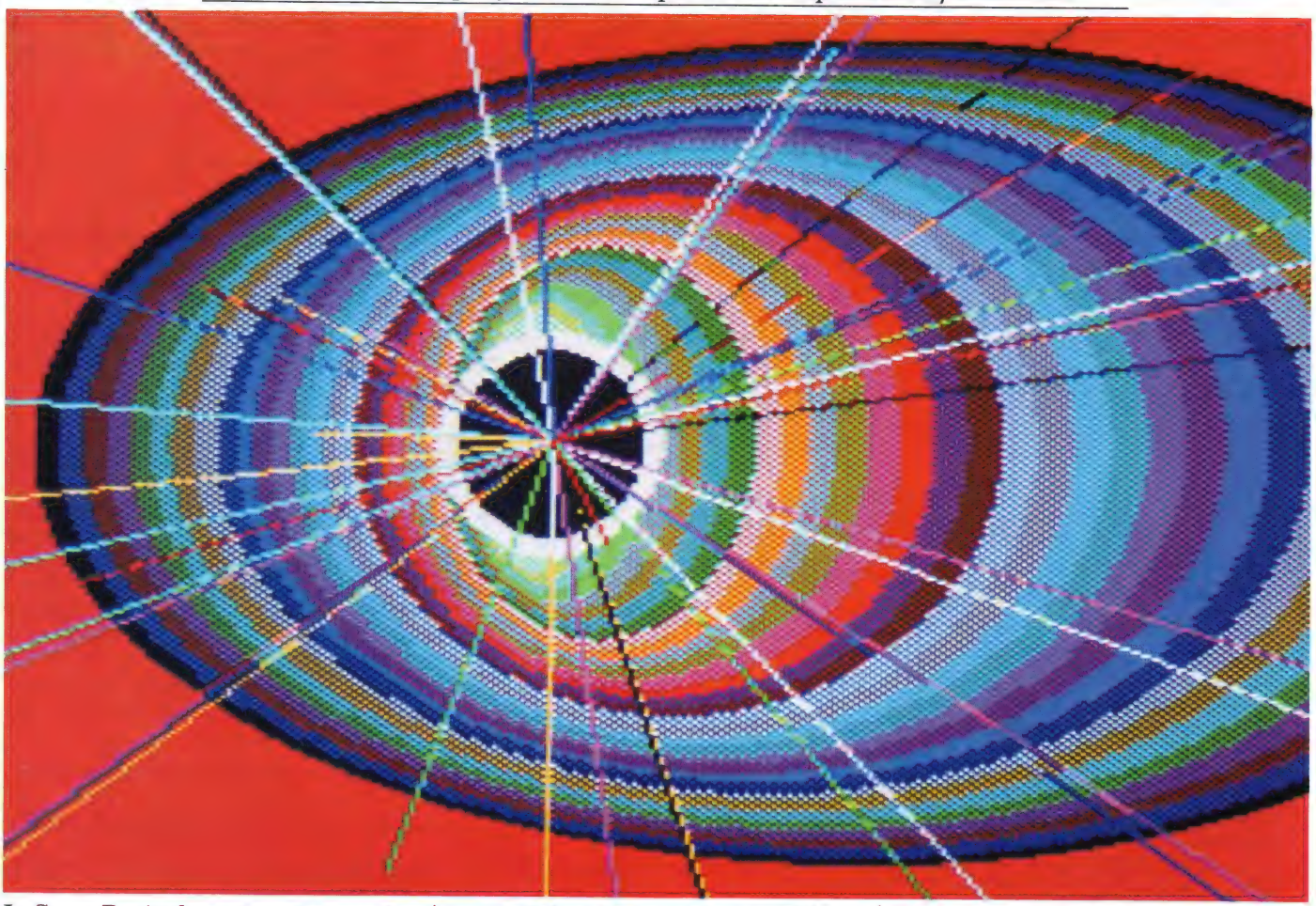
Britain's two most popular personal computers, the Commodore 64 and Spectrum are covered from basic to semi-expert in **Which Micro Magazine's Handbooks.**

Author Pete Gerrard, a regular columnist for Which Micro, has put together an accurate and practical guide to both computers, at £4.99 per book.

Many programs are included and both 160-page books are spiral bound for easy use and are available in all good bookshops from November or direct through your letterbox by sending to EMAP Books, Bushfield House, Orton Centre, Peterborough PE2 0UW.

TROOPING THE COLOUR

Making up for the shortcomings of SuperBasic as a tool for the games programmer, Andrew Cryer explores the mysteries of the QL's screen display and develops two indispensable functions.

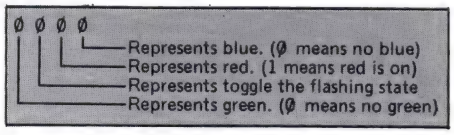


In SuperBasic there is no instruction to read the colour of a pixel at a point on the screen. This article shows how to do just that. It provides two functions **colour_eight** and **colour_four** which return the colour number of a point on the screen in the eight-colour and four colour mode. The point is specified in terms of its pixel co-ordinates as I don't, as yet, know of a method of reading the current scaling factor and origin that the QL has been set to. (Anyone any suggestions?)

Before explaining the routine to read the colour of a pixel, one should examine how the colour of a pixel is stored in memory.

The screen occupies 32K bytes of memory, from address 131072 (the top left-hand corner) to address 163839 (the bottom right-hand corner). This memory is divided up into 256 blocks, each block representing one row of pixels across the screen. Thus a total of 128 memory locations are left to represent a single horizontal row of pixels. Since each memory location holds a binary number (all 0's and 1's) with 8 bits in each, this gives a total of 128*8 (or 1024) bits to represent a horizontal row of pixels.

The way that the colour of a pixel is stored in memory differs slightly in the two modes. Let us consider mode 8 first. This mode allows eight different colours, each one made up from combinations of red, green and blue. In addition, a pixel may be flashing or steady. The four options (red, green, blue and flashing) can be represented in terms of the four bits in a binary number, where each bit may be 0 or 1, to indicate whether the option is off or on. Initially, all the bits will set to zero.



The flash bit is different from the others in that it affects that pixel and the rest of the line. It toggles the flash state either on or off. So if flash is currently off, setting the flash bit would toggle it on, and vice versa.

With 1024 bits available to represent a horizontal row of pixels, and ▶

with four bits needed for each pixel, the QL can have 1024/4 (or 256) pixels in a line across the screen in mode 8.

In mode 4 a pixel can only have four colours, each one made up from various combinations of just two colours: red and green. The state of these two colours can be represented by a two bit binary number (shown below for green).

```
Ø 1
└──Represents green. (1 means green is on)
└──Represents red. (Ø means no red)
```

With 1024 bits available to represent a horizontal row of pixels, and with only two bits needed for each pixel, the QL can have 512 pixels on any line across the screen in mode 4.

Coded Information

The screen information is stored in a coded form, with the bits for any one pixel shared between two successive memory locations. (This is probably because the QL's 68008 processor is most efficient when addressing 16 bit numbers.) To show how the information is stored, let **G, F, R,** and **B** represent green, flashing, red and blue. Let 1, 2, 3 etc etc represent successive pixels across the screen.

Consider mode 8 first. In mode 8, 4 bits of information are required for each pixel: one bit for green, one for flashing, one for red and one for blue. The format and ordering is as shown below.

High byte (Even address)								Low byte (Odd address)							
G1	F1	G2	F2	G3	F3	G4	F4	R1	B1	R2	B2	R3	B3	R4	B4

Colours in a pixel are built up by setting the appropriate bits. So the range of colours which can be built up in mode 8 (where t-flash is short for 'toggle flash), set out in a table would be:

Colour number	Colour	Bits set
Ø	Black	None
1	Blue	Blue
2	Red	Red
3	Magenta	Blue + Red
4	Green	Green
5	Cyan	Green + Blue
6	Yellow	Green + Red
7	White	Green + Blue + Red
8	Black, t—flash	+ t—flash
9	Blue, t—flash	Blue + t—flash
10	Red, t—flash	Red + t—flash
11	Magenta, t—flash	Blue + Red + t—flash
12	Green, t—flash	Green + t—flash
13	Cyan, t—flash	Green + Blue + t—flash
14	Yellow, t—flash	Green + Red + t—flash
15	White, t—flash	Green + Blue + Red + t—flash

High byte (Even address)								Low byte (Odd address)							
G1	G2	G3	G4	G5	G6	G7	G8	R1	R2	R3	R4	R5	R6	R7	R8

In mode 4, 2 bits of information are required to represent a pixel: one for red and one for green. The format and ordering of this is shown above.

So for mode 4, the range of colours is more restricted than for mode 8. It can be regarded as being built up in the following fashion.

Colour number	Colour	Bits set
0	Black	None
1	Black	None
2	Red	Red
3	Red	Red
4	Green	Green
5	Green	Green
6	White	Red + Green
7	White	Red + Green

Spot Colour

Reading the colour of a pixel involves two operations. Firstly identifying the two bytes used for storing the colour of that pixel, and secondly decoding the appropriate bits of those bytes to give the colour number. Both operations can be incorporated into a single function which will return the colour number of a pixel at a specified point. Because mode 4 and mode 8 use a different format, it is easiest to write separate functions for each mode.

The function for mode 8 is:

```
1000 DEFine FuNction colour_eight (x,y)
1010   LOCal col, memory, high, low
1020   memory = 163712+ (x DIV 8) ★2-y★128
1030   high = PEEK (memory) DIV (4^(3-(x DIV 2) MOD 4))
1040   low = PEEK (memory + 1) DIV (4^(3-(x DIV 2) MOD 4))
1050   REMark Separate off just the blue and red bits
1060   col = low && 3
1070   REMark Add the green bit
1080   col = col + (high && 2) ★2
1090   REMark Add the flash bit
1100   col = col + (high && 1) ★4
1110   RETurn col
1120 END DEFine
```

The origin (0,0) is taken as the bottom left-hand corner of the screen. This means that the function may be used to test a pixel at, say, the centre of the screen by entering a line such as:

```
PRINT colour_eight (128,128)
```

The address of the high byte is calculated using the formula in line 1020. Here x is the x co-ordinate of the pixel, being in the range 0 to 511, and y is the y co-ordinate of the pixel, being in the range 0 to 255. Lines 1030 and 1040 read the two binary numbers from memory and shift them to ensure that their two least significant bits represent the colours of the pixel concerned. Lines 1060, 1080 and 1100 construct a binary number representing just the colour number of the pixel. Line 1110 returns the complete colour number.

The corresponding routine for mode 4 is as follows:

```
1200 DEFine FuNction colour_four (x,y)
1210 LOCal col, memory, high, low
1220 memory = 163712+ (x DIV 8) ★2-y★128
1230 high = PEEK (memory) DIV (2^(7-(x MOD 8)))
1240 low = PEEK (memory+1) DIV (2^(7-(x MOD 8)))
1250 REMark Take the red bit
1260 col = low && 1
1270 REMark Add the green bit
1280 col = col + (high && 1) ★2
1290 RETurn col★2
1300 END DEFine
```

The functions are used in the same way as the SuperBasic's standard functions.

Printer Dump

As just one example of their application, I now give a printer dump routine for EPSON printers type MX80, RX80 and FX80. The printer

has to be one fitted with a serial interface, suitable for the QL. Alternatively one of the currently advertised printer interfaces could be used.

The routine is listed below for use in mode 8, with colour_eight (x,y) testing points on the screen. However by replacing the colour_eight function in line 1520 with the colour_four function, displays in mode 4 can also be dumped. All colours appear as black. There are no shading effects.

When the Epson printer is in the dot-addressable mode, it allows each of the eight dot-producing wires in the print head to be addressed separately. The dump routine scans across the screen, examining a strip eight pixels high. It then converts the pixel colours into codes to be sent to the printer and thus produces a one-to-one image between pixels on the screen and dots on the Epson. Each column of eight pixels is converted to a binary number of 1s and 0s, 1s corresponding to light areas and 0s to dark areas. The eight bit binary number is then sent to the printer. As I have arbitrarily selected all output to the printer to appear on channel 5, the code is sent to the printer by the instruction:

```
PRINT #5,CHR$(char);
```

The process continues until the end of the line has been reached. It then commences for the next line, until the whole screen has been interpreted.

The routine is called 'dump'. Here is its listing:

```
1400 DEFine PROCedure dump
1410   LOCal x,y, height, char
1420   REMark Set up printer for
       dump
1430   OPEN #5, ser1
1440   BAUD 4800
1450   PRINT #5, CHR$ (27);
       CHR$ (65); CHR$(8);
       CHR$(27); CHR$ (50);
       CHR$ (10);
1460   FOR y = 248 TO 0 STEP -8
1470     REMark Set up printer
         for 1 line
1480     PRINT #5, CHR$ (27);
         CHR$ (76); CHR$ (0);
         CHR$ (2);
1490     FOR x = 0 TO 511
1500       char = 0
1510       FOR height = 0 TO 7
1520         IF colour_eight (x,y
             + height) >0 THEN
             char
             = char+2^height
1530       NEXT height
1540     PRINT #5, CHR$ (char);
1550     NEXT x
1560     PRINT #5, CHR$ (10);
1570   NEXT y
1580   CLOSE #5
1590 END DEFine
```

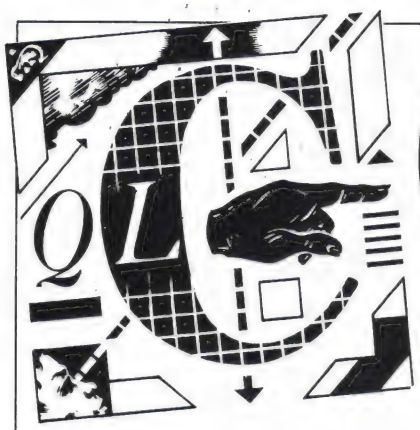
The outer FOR . . . NEXT loop scans down the screen, one line of eight dots at a time. The next loop scans along each line, while the inner loop scans down any one column of dots to calculate the binary number. Line 1540 then sends this to the printer. The codes sent to the printer in lines 1450 and 1480 are necessary in order to prepare the printer for the dump.

The dump routine will produce a copy on the printer of those colours on the screen which have a colour number of greater than zero, ie all colours other than black. It can be changed so as only to dump specified colours by altering line 1520. For example, for the printer to reproduce only colours 1 (blue) and 6 (yellow), line 1520 would need to be replaced with:

```
1520 IF colour_eight (x,y +
     height) = 1 OR colour_eight
     (x,y + height) = 6 THEN
     char = char + 2^height
```

The dump routine produces a true one-to-one correspondence between pixels on the screen and dots from the printer. This tends to produce some distortion since the spacing between pixels on a television screen is different from that produced by the printer. You may like to see if you can improve matters by playing around with some scaling. Try adding a supplier to an appropriate co-ordinate in line 1520.

As it stands the routine is slow. So plan your dump for over lunch!



C SERIES

More vital information for eager C enthusiasts, compiled especially for QL User by Peter Rodwell.

In our last episode we gave you an express journey through the basics of C language. Now we'll elaborate on these and introduce some new aspects of the language.

We saw that in C a program comprises a number of separate modules, called functions. One of these must be called **main** and denotes the starting point of the program. It doesn't matter where you put this start module in your program — you can put it at the beginning, the end or anywhere inbetween — the compiler will find it.

The remaining modules are roughly equivalent to BASIC subroutines; typically, a C program's main module will comprise mostly calls to functions, which in turn will include calls to other functions . . . and so on. The first step in writing a C program (and, for that matter, a program in any structured language) is to break the job down into its component parts and then to write each one as a small module.

One very pleasant result of programming like this is that you can retain a clear mental image of what is going on all the time. If, for instance, you have ever written a really long BASIC program, the chances are that eventually you will have stumbled across lines like

12030 GOSUB 20550 : GOSUB 1200 : GOTO 14100

There is no way of telling from this what is going on: the GOSUBs give no indication of what those subroutines do, or which variables they affect. C allows you to give each function a meaningful name and when you use it, you must include in its call the names of the entities it requires:

```
a = total_of(x,y,z);
```

Let's write **total_of,** the function which adds three numbers together and returns the result. Here's one version:

```
total_of(x,y,z)

int   x, y, z;
{
int   sum;
sum = x + y + z;
return ( sum );
}
```

By now you should be able to follow this quite easily. Directly after the name of the function, we have three numbers which are to be added together. On the next line, we state exactly what these are: integers. Inside the function, we declare another variable, **sum,** which we then use to store the result of adding three numbers, before returning it.

Using Functions As Variables

It's important to realise that, unlike most BASICs, C regards all the variables used within a function as being *local* to that function unless you decide otherwise. For instance, in the function **total_of,** we have a variable called **sum.** We might in fact use this variable name in other functions in the same program – it doesn't matter. When **total_of** is used, a temporary variable, **sum** will be created automatically, just for this function; it will be discarded when we leave the function. Thus, any other variables called **sum** elsewhere in the program will remain unaffected.

C also allows us to use a function call wherever we can use a variable. Here's a program which illustrates this:

```
main()
{
int   x, y, z;

x = 10;
y = 20;
z = 30;
printf ("Total = %d/n", total_of
(x,y,z));
}

total_of (first_number, second_
number, third_number)
int   first_number, second_
number, third_number;
{
return (first_number + second_
number + third_number);
}
```

Well, there are several points to notice here. Firstly, in our **main** function, we have used the call to **total_of** as though it were a variable, placing it directly within the call to **printf.** But the major changes occur within **total_of** itself.

Most obvious is that within the function, we refer to the three numbers by different names. This doesn't matter at all – the idea is that you choose the names most appropriate to that function. After all, we may want to use **total_of** several times within a program, using different variables, so within the function itself we refer to the integers by names which make sense *within that function.*

We have also disposed of the temporary variable **sum.** All the work is done within the **return** statement instead. This is not only quicker to write and execute but illustrates neatly the brevity which makes C programming such a pleasure.

Earlier we said that all variables are local to functions unless we decide otherwise. There is a way of creating these *global* variables: you just declare them outside any function. To access them from within a function, you simply declare them as *external,* like this:

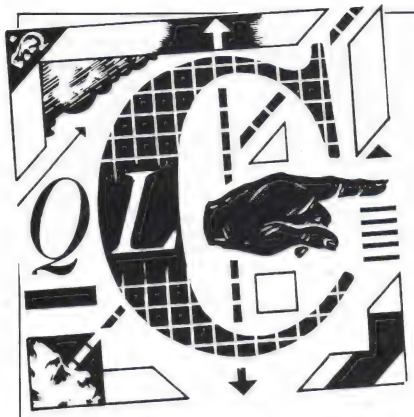
```
int   sum;

main()
{
int   x, y, z;
extern int   sum;

x = 20; y = 30; z = 40;
total_of (x,y,z);
printf ("Total = %d/n" ,sum);
}

total_of (first_number, second_
number, third_number)
int   first_number, second_
number, third_number;
{
extern int   sum;
sum = first_number + second_
number + third_number;
}
```

This is a much more cumbersome way to carry out this particular operation – unless you need to access ▶



sum from other functions within the program. Note the declaration **extern int** in both **main** and **total_of.**

If you're really on the C ball by now, you should be wondering about this business of using functions as variables. If all variables need to be declared before they can be used, shouldn't functions also need to be declared?

Well, yes, strictly speaking you should declare functions just as though they were external variables. In practice, most compilers will allow you to get away with not declaring functions which return integers. Most however will insist that you do declare functions returning other entities, such as **char**acters.

And the declarations must be made exactly as for an external variable. The following example shows a simple function which takes a character typed in at the keyboard, converts it to upper case if it's lower case, and prints it on the screen.

```
main()
{
char   c;
extern char   getchar(), to_
upper();

c = to_upper (getchar());
printf ("%c/n",c);
}

char to_upper (c)
char   c;
{
return ( (c > 'Z') ? c – 'A' : c );
}
```

Here we can plainly see the declarations. In **main** we have two functions to declare, both of which return characters. One is the standard C character input function **getchar,** which usually gets a character typed in at the keyboard. The other is the case conversion function, **to_upper,** which we've written ourselves. And when writing **to_upper,** we start off by declaring it to be a function which returns a character, simply by putting **char** at its beginning. Notice that **to_upper** uses the alternative conditional expression described in the last issue. Can you work out what this is doing here?

Conditional Tips

Conditional expressions – whether in abbreviated form or using **if** – are always evaluated to one of two values, true or false; in the computer, there are represented internally as 1 and 0 respectively. We can capitalise on this to make conditional expressions shorter and easier to use.

Consider the following examples:

```
if ( x == 0 )
while ( y > 1 )
```

If we are certain that we will always be dealing with quantities which are either zero or not zero, we can abbreviate these:

```
if ( !x )
while ( y )
```

The first uses the unary ! ("not") opertor and you can think of it as *"if not x"*. The second is equivalent to "while y has any non-zero value" or "while y exists".

ILLUSTRATION BY JEFF FISHER

Multiple Choices

One common requirement in programming is to take one of several courses of action depending on the outcome of a calculation. In BASIC this can be handled in two ways.

```
100 IF X = 1 GOTO 5000
110 IF X = 25 GOTO 6000 etc.
```

In C, too, we can use multiple **if** statements to achieve precisely the same affect. Basic also allows a crude switching operation with ON...GOTO and ON...GOSUB. C provides a much more powerful switching device using the **switch** facility. This allows us to evaluate an expression and then take appropriate courses of action:

```
switch ( x )
  {
  case 1:  c = to_upper (c);
           break;
  case 27: c = getchar (c);
           break;
  default: printf ("This is the
           default option/n");
           break;
  }
```

switch (x) is where we establish the criterion for the switching operation. You can put a simple variable in the brackets, as we have done here, or a complex expression, perhaps incorporating one or more function calls. The only restriction is that it must finally become an integer.

We then define the different courses of action with **case** and the appropriate integer value which triggers each choice. You can include any number of program statements within each **case** – but note the use of **break.** Taking the first option, **case 1:** as an example, this will only be executed if x = 1; if we omitted the **break** here, though, the program would "drop through" into the next bit, which we only want to execute if x = 27. You can, though, make use of this feature:

```
case 4:
case 5:
case 6: printf ("x = 4, 5 or 6/n");
        break;
```

This construction allows you to execute the same piece of code for a range of values.

With **switch** we can also specify a **default** course of action to be taken if none of the **case** conditions are fulfilled. The use of **default** is entirely optional. If you miss it out, then if none of the **cases** can be executed, the program will ignore the lot of them and continue with whatever comes after them. One handy tip: strictly speaking, we needn't have put a **break** after the default part above, as the program would automatically drop out of the **switch** in any case. However, it's very easy to add another **case** after the default one and forget to add the requisite **break** at the end of the default option, which could lead to some interesting debugging experiences later. As a matter of habit, end *every* **case** with a break, just to be safe.

Another Loop

We have already seen two types of loop – **for** and **while.** Both of these place the condition governing the number of times the loop is executed at the top of the loop. C provides a third type of loop – the **do** loop – in which the condition is placed at the bottom.

To understand the significance of this, consider the following:

```
x = 1
while ( x < 1 )
  {
  printf ("x is zero/n");
  }
```

Because x has been set to 1, the **while** loop will never be executed and we will never see the **x is zero** message. Now look at this:

```
x = 1;
do{
  printf ("x isn't zero/n");
  } while ( x < 1 );
```

Here, the condition is not encountered until *after* the first pass through the loop. Thus, using the **do...while** construction ensures that the body of the loop will *always* be executed at least once. The difference is subtle but can be of great importance.

Hip, Hip, Array

Most BASIC programmers will be familiar with the concept of arrays: a table of numbers, characters or alphanumeric strings. C also has arrays, and with one important exception they are very similar to those in BASIC.

The exception concerns strings. C has very limited string-handling capabilities; the only thing you can do with a string directly is to print it:

```
printf ("This is a string");
```

Instead, C treats strings as arrays of single characters, which is on balance a mixed blessing. It does make a lot of string operations very easy – thus making C a good language in which to write word processors – but it can require some intricate programming at times.

First, though, let's look at how arrays themselves are handled. Just as in BASIC you must declare the size of an array. In C, you also have to declare the type of objects it holds:

```
int  table[20];
```

declares an array called **table** containing 20 cells or rows of integers, while

```
int  grid[10][3];
```

declares a *two-dimensional* array of integers made up of 10 rows, each of three columns. A string is declared as an array of characters:

```
char  line[80], screen[25][80];
```

would give us two arrays, one a single line holding up to 80 characters and one holding 25 lines each of 80 characters. Note that some mental gymnastics are required here: we use slightly different terminology with string arrays simply because this makes it easier to envisage what's going on. In fact they're not treated any differently by the computer – it's purely to make things easier for us.

Accessing the elements in an array is just as you would expect if you've had some BASIC experience:

```
table[8] = 27;
grid[4][2] = 35;
etc.
```

You can of course put variables or even functions which return an integer between the square brackets to access a particular element. This allows you to use **for while** or **do** loops to assign values to elements in an array.

68020 THE CHIP

If you thought you'd heard the last of QL's 68008 32/8 microprocessor then you're right, though probably for the wrong reasons. As Adam Denning points out, compared with the 68020 it dwindles into insignificance.

We have all heard enough of Motorola's 68008 central processing unit that's inside the QL, and of the long-running debate surrounding the architecture of this chip. It's a 32 bit processor, true enough, but it only has an eight bit data bus through which to communicate with the outside world. This relatively small data bus makes the chip rather slower than its more expensive brethren – the 68000 and 68010 – but it does make it a lot easier to interface the standard 6800 family series of peripheral chips like the 6821 PIA and the 6850 ACIA. A PIA is a 'peripheral interface adapter' and an ACIA is an 'asynchronous communications interface adapter', both of which are very useful for adding serial ports, printer ports and so on to a more basic computer set-up.

The Chips Are Down

For quite a while now Motorola has been promising the ultimate member of the 68000 family, the 68020. This was to be fully 32 bit in that the program counter, the data bus and the internal architecture would all be capable of supporting full 32 bit quantities. Now the chip is here, we find it fulfills rather more than just these expectations.

Just about every undesirable feature of the previous 68000 family has been removed, much to the relief of the 68000 assembly language programmer, and a whole host of additional facilities have been added. Motorola claims that 51 new addressing modes have been added. An addressing mode is a method of accessing data used in the assembly language of a particular processor, and the 68008, 68000 and 68010 are well endowed with various modes already. The earlier members of the family have one large problem, though – they are not **orthogonal.** This means that there are some irregularities in the instruction set, such as the ability to read from 'Program Counter Relative' locations but not write to them, and the strange lack of uniformity in the data sizes useable by certain instructions. While some, such as CLR, are capable of operating upon bytes, words and long words, others, such as CHK, can only act on word-sized data. There is no good reason for this, and Motorola is obviously aware of it.

Caching In

We will examine the differences in the 68020 from a programming point of view as there are unlikely to be that many people with either the money or the time to build a 68020 system just yet! Taking the internal structure first, we notice that it has the same number of address and data registers as the other members of the family (A0 – A6 and D0 – D7) but now has three different A7 stack pointers. The user stack pointer remains the same and the supervisor stack pointer becomes, in effect the interrupt stack pointer. The third stack pointer, the master stack pointer, is there to facilitate multi-tasking, where each process can have its own small stack to hold information and data specific to that task.

More interesting are two other new registers, the CACR (cache control register) and the CAAR (cache address register). The cache is an on-chip area of RAM devoted to holding assembly language instructions. By holding the next few instructions actually in the processor (and the 68020 can hold 256 bytes), processing time can obviously be decreased dramatically as there are correspondingly fewer slow memory accesses, or rather the memory accesses are done at non-critical times.

The CACR is used to enable and disable the cache, and to clear a single entry or the entire cache, while the CAAR is used when a single cache entry is being cleared to point the entry in question.

One more added register is the vector base register (VBR). On most earlier members of the family (aside from the 68010, exception vectors are always in fixed positions within the memory map. This is a physical constraint which no amount of software can remove. This means that to handle exceptions on the 68000 or 68008, memory addresses from 0 to the top of the exception vector table must be present. On the 68020 this is not so – the VBR holds an address which points to the first entry in the exception vector table. The exceptions are roughly equivalent to less sophisticated processors' interrupts, except that more are generated by software. So, multifarious interrupts, illegal instructions, division by zero, registers out of bounds, traps and all other exception generators can now be directed by only one level of indirection to any location in the processor's memory map.

Addressing Down

Addressing modes are of course one of the most important aspects of a new processor as far as the programmer is concerned, so let's look at a few of the 68020's. The important concept in 68000 assembly language when dealing with addressing is the "effective address". This is the address that a particular addressing mode eventually yields as the source of its data or as the destination of its result. Take the simplest case, *absolute* addressing:

MOVE.L address,A3

Here the effective address is the value assigned to the lable **'address'** by the assembler – usually through an EQUate by the programmer. So, if **address** was $20000, then the effective address is $20000. Thus, register A3 is loaded with the long word contents of this address. If we examine the addressing modes of standard 68000 processors, and work out their effective addresses, we can see the additions in the 68020 more clearly. We will ignore *immediate, implied* and *register direct* as these are obvious and have no scope for alteration.

1.—*Address register indirect:*

MOVE.L (A1),D1

Effective address is the memory location whose address is in A1. So, if A1 held $20000 and location $20000 held $12345, then D1 would be loaded with $12345.

2.—*Address register indirect with pre-decrement:*

MOVE.L D1, – (A7)

Effective address is the same as above, except that the value in the

address register is decreased by four before the EA is used. If the data size being moved were .W, then the EA would be (A7 – 2), and if the size were .B, the EA would be (A7 – 1).

3.—*Address register indirect with post-increment:*

MOVE.W (A7) + ,D0

Effective address is (A7), but the data size (two bytes here) is added to the address register after the EA has been used.

4.—*Address register indirect with displacement:*

MOVE.B disp(A6),D4

Effective address is (A6) + disp, with **disp** being a 16 bit two's complement quantity.

5.—*Address register indirect with offset and displacement:*

MOVE.W disp(A6,A1.L),Dt

Efffective address is (A6 + A1) + disp, with **disp** being an 8 bit two's complement quantity.

In all except pre-decrement and post-increment, the base address register can be replaced by the program counter, thus making code position independent. A restriction here is that data can only be read using such modes – it cannot be written.

The 68020 adds to these modes. First, there is the ability to bring in a further 'outer' displacement, which is added to the effective address calculation either before a memory indirection (pre-indexing) or after (post-indexing). So, we may have:

MOVE.L ([base,A3],D4.L,disp),D2

Here the effective address is (A3 + base) + D4 + disp. The parentheses around the (A3 + base) refer, of course, to indirection. This post-indexing thus becomes equivalent to:

```
MOVE.L BASE(A3),D2
ADD.L  D4,D2
ADDI.L #disp,D2
```

Pre-indexing:

MOVEA.W ([base,A2,A5.W],disp),A3

The effective address here is (A2 + base + A5) + disp, making it like:

```
MOVE.W base(A2,A5.W) ,A3
ADDA.W #disp,A3
```

These addressing modes are further enhanced by the ability to add a scaling factor of 1, 2, 4 or 8 into the calculation. Here the address or data register forming the index register is multiplied by the scaling factor before it is used in the effective address calculation. Note that the index register itself is not altered, just its value in respect to the EA calculation.

There are quite a few more addressing modes available on the 68020, but let's leave them and look at the enhanced instruction set. We now have conditional traps, like

TRAPNE #3, and TRAPCS #1

and various checks and compares which do two things in one go. Multiplication and division are improved to allow most data sizes to be manipulated, and the **PACK** and **UNPK** instructions allow compact handling of BCD numbers. High level modules (rather like procedures) can be called and returned from using CALLM and RTM and there are a whole host of co-processor instructions to handle memory management, floating point processors and so on. Virtual memory, where discs or whatever hold the main bulk of a program and only the relevent section is present in RAM at a time, is also implemented. Varying data sizes are taken care of via the bit field extension. A bit field is a user defined number of bits, from 0 to megamillions (yes, literally!), starting and ending at defined places. We could, for example, specify the third, fourth and fifth bits in a data register as a bit field, or the entire QL screen. Now think of that!

There is of course a whole lot more to the 68020, but this article should have given us all a fair taste of its capabilities. All we need to do now is wait 28 Sinclair days for the QL+!

COMPETITION

Questions and Answers

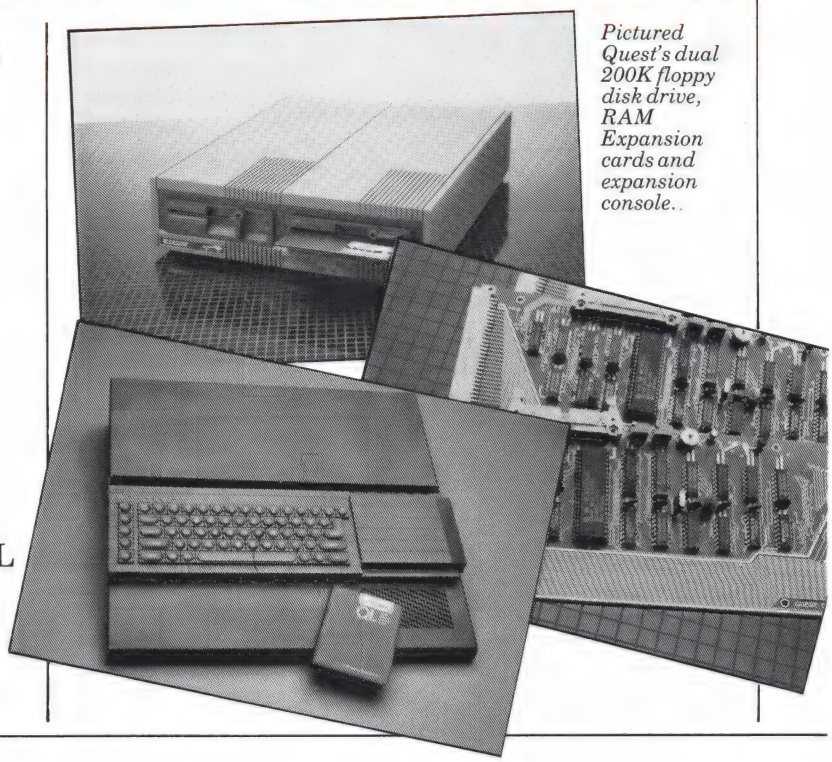
1st Prize: A 200K floppy disk drive, complete with CP/M disk

2nd Prize: A 128K RAM card and the Tally Accounts package

3rd Prizes: Three Expansion Consoles

(Part of the exciting range of products in the new Quest 'QL executive series')

Here's a great way to get a head start in the QL stakes. Enter our competition, organised in conjunction with Quest Automation, and you could be well on the way to owning the latest QL hardware.



Pictured Quest's dual 200K floppy disk drive, RAM Expansion cards and expansion console..

COMPETITION

Here's what we said about the range in the December/January issue:

"Right on schedule, Quest have released their range of disk systems and RAM extensions.

Firmly establishing themselves as the biggest peripherals maker for the QL, Quest unveiled Shugart 5.25" floppy drives of 200K, 400K and 800K capacities, with corresponding prices of £249 to £599.

And Quest have leaped into the increasingly competitive RAM extension market with a range of devices offering 64K (£99), 128K (£159), 256K (£299) and 512K (£499). The two product ranges will be important for liberating the QL from its untrustworthy microdrives. The RAM extensions will relegate microdrive access to an initial booting up and final storage of files, while access to disk drives will please potential business users of the machine.

Speaking of which, Quest have released a QL accounting suite called Tally. The program will run under both CP/M 68K and QDOS, and consists of Sales Ledger, Sales Invoicing and Stock Control, and costs £99. A version called Tally II, costing £50, covers Purchase and Nominal Ledger.

Modules integrate not only with themselves but with all four Psion progs, and if the Stock Control package is any indication then Tally should be something special."

HOW TO ENTER:

All you have to do is answer the questions below, which are based on information taken from the Quest folder inside our December/January issue.

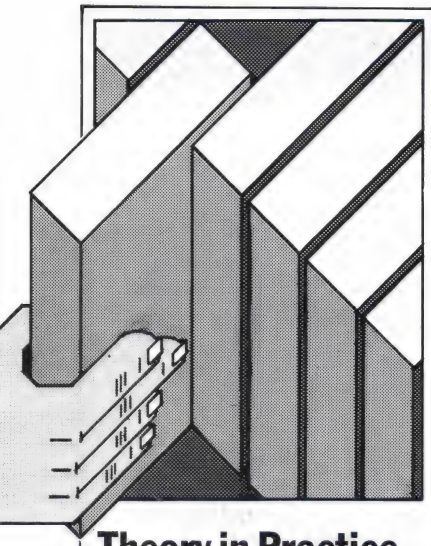
All the correct entries will then be judged on the following (to be answered in 10 words or less): We called last month's news story 'Quest For The Engram'. Why is that an appropriate title?

SEND YOUR ENTRIES

(any not on a postcard will be disqualified)
TO: Questions and Answers Competition,
QL USER, Priory Court, 30-32 Farringdon Lane,
London EC1R 3AU.

QUESTIONS:

1. The Quest expansion boards are 'based on the latest 64K D RAM modules'. What does the accronym D RAM mean?
2. Access to RAM contents using these boards is quoted at 150ns. What does 'ns' represent?
3. Quest claim exclusive rights on 'CP/M 68K'. What does CP/M stand for?
4. What is the formatted storage capacity of the Firefly Winchester disk?



BOOKMARKS

Nicky Trevett reviews the latest offerings for the QL – from serious applications to games.

Theory in Practice

It makes a refreshing change to find a book that actually contains what it says it does on the cover – *a guide to the QL* – even if the title itself is a little misleading. *Quantum Theory* by Jeremy San, Fouad Katan and Simon Rockman (Century Communications Ltd) is really quite the opposite, more Quantum in practice.

Priced at a very reasonable £5.95, it provides a down to earth, lively and prctical guide to getting to grips with the QL machine itself, as opposed to the four programs that come with the QL (which barely get a mention). The emphasis is all on learning to use the SuperBasic programming language, and via SuperBasic all the QL's inherent facilities, like graphics, sound, arithmetic and the in-built clock.

Everything is approached from the software angle; you are told, for example, how to format the microdrives, call up a list of files and delete them, but not much about how microdrives themselves work. Similarly, there is a look at the way the operating system QDOS (affectionately referred to as Domesdos because it "kills all known bugs") handles hardware devices, but nothing about the inner workings of the microprocessor.

There's a lot of good, solid advice, too. You are warned about the likelihood of crashing the system when attempting to run machine code, for example, and about the perils of leaving microdrive cartridges in the QL when experimenting with just about anything which could result in a machine reset, clearing RAM and probably the tape as well. There's also a conscious effort to avoid jargon and dry technicalities; the book is quite likely to tell you to skip a chapter if the topic frightens you, and whenever it introduces a worrying new concept like structured programming, it comfortingly tells you not to bother your head about it because it's always simpler than it sounds.

A very accessible, useful and humorous book which can be recommended.

Hard Work

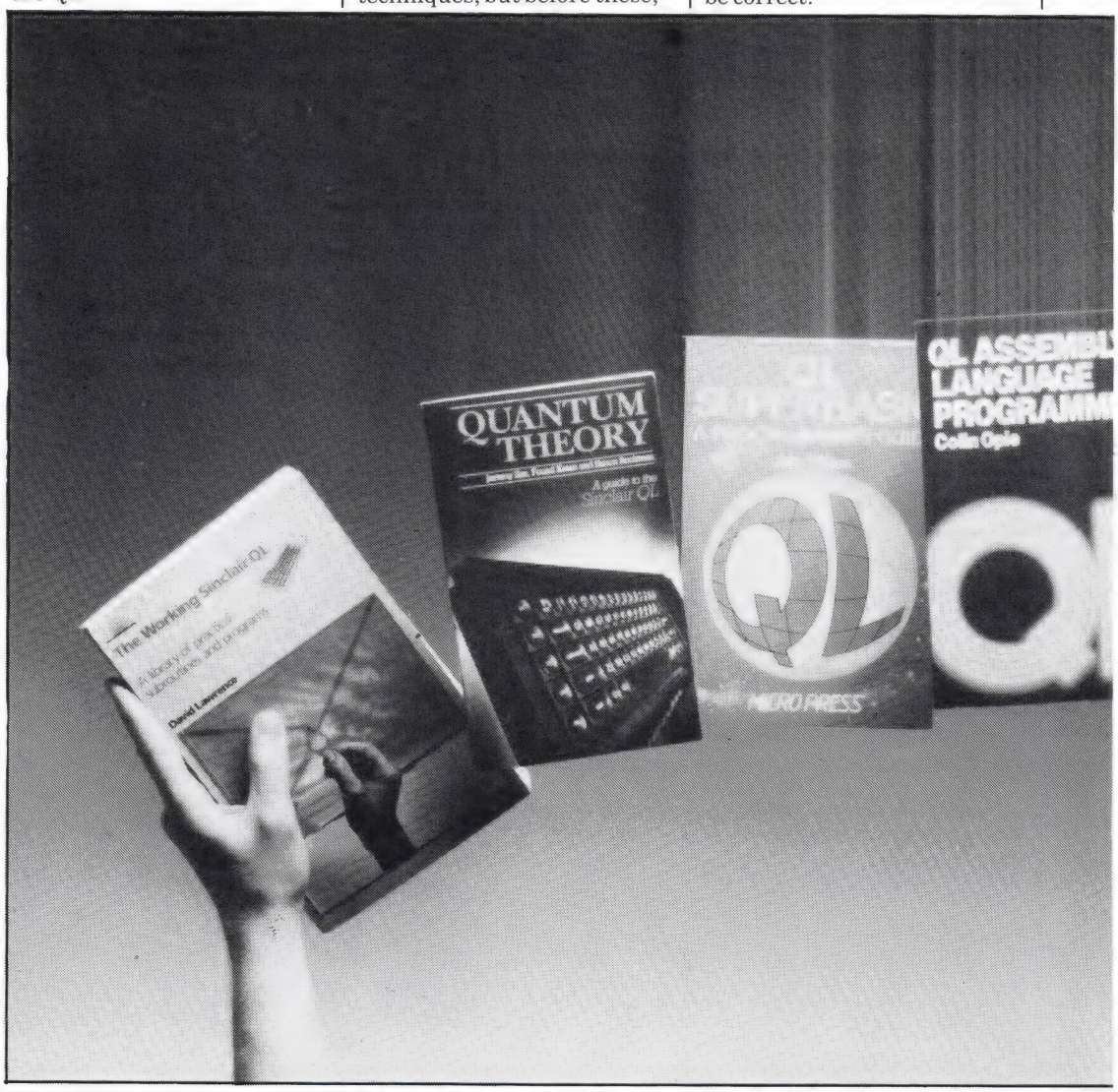
The Working Sinclair QL by David Lawrence, published by Sunshine Books at £6.95, offers something rather different.

It is another in Sunshine's 'Working' series which provides programs, subroutines and an introduction to more advanced programming techniques to enable users of a variety of microcomputers to make better use of their machines. Now it's the turn of the QL.

Here there are programs and routines grouped into four main sections: experimenting with time and clocks; various graphics and sound routines; creation of personal filing and storage systems; and programs for keeping a record of a bank account, for keeping 'traditional' accounts and for managing a family budget.

The author says his book can be used in a variety of ways – as a collection of useful programs to adapt to your own purposes; as a library of subroutines to use as a basis for your own programs; and as an introduction to SuperBasic programming. I would not argue with the first two, but would certainly not recommend the book as an introduction to programming. There is a great deal to be learnt about SuperBasic programming techniques, but before these, and the routines themselves, can be adapted, modified and put to use, you would need a solid grounding in basic programming.

One particular point of note is the somewhat daunting-sounding "Sunshine Checksum Generator" provided at the end of the book. This is simply a means of checking that you have keyed in the listings correctly, but looks quite alarming. To use it, you have to load the Checksum program into memory, place the cartridge containing the program to be checked into drive I, supply the program name and the start and finish lines. The Checksum tots up the values of the characters in each line of the program you have entered, which you then check with the totals provided in tables in the book. If the sums of the values match, your program should be correct.



On the whole, I think it would be simpler to take care to enter the programs properly in the first place!

Advance Beyond GOTO

Would-be SuperBasic programmers must already be bewildered by the choice of books available to them already, but still they come. *QL SuperBasic – A Programmer's Guide*, by John Wilson, published by Micro Press is a slim volume which feels expensive at £6.95. all the more so since it is rather limited in scope.

It is no course in BASIC programming, or even in SuperBasic programming. It assumes a fundamental knowledge of BASIC, asks you to read the book in conjunction with the QL User's Guide, and warns at the start that it does not even cover every SuperBasic keyword.

But what it does do, quite efficiently, is give the programmer a good understanding of the ways in which SuperBasic far exceeds ordinary BASIC, especially in the way it is structured. It is all about advanced and superior programming, replacing outmoded GOTO and GOSUB statements with much more elegant keywords.

After a lengthy look at program structure, it covers ways of manipulating the QL display, sound, timing and filing, and provides a large program at the end for practising the new techniques introduced throughout the book. There are sample listings throughout, but these are mostly short and to the point, provided only to illustrate the programming technique under discussion.

There is a technical feel to the text, which is full of programming jargon and the sort of phraseology more commonly found in commercial programming departments, like "top down design" and "bottom up assembly". But this should not worry the experienced BASIC programmer who seriously wants to get to grips with SuperBasic and write better programs.

Mammoth Assembly

However, only those programmers already familiar with 68000 assembly language programming can hope to make much sense of Colin Opie's *QL Assembly Language Programming*, published by McGraw-Hill at £12.95. This is a mammoth work, written with the assistance of the QL QDOS designer (Tony Tebby), which is all about assembly language programming on the QL, as opposed to assembly language programming in general. It is intended to be used in conjunction with McGraw-Hill's program editor/68000 assembler package, price £29.95.

There are no concessions made for the hobbyist writing programs for fun. This is advanced, technical material for those already immersed in the subject, like "educational and training institutions" and "OEM design engineers". The book covers the 68000 processor itself with its instructions and addressing modes; QL system procedures, including the SuperBasic machine code commands; provides a number of chapters full of example programs; and finishes with a look at the editor/assembler package you are assumed to have bought.

The useful appendices include a 68000 instruction set summary, a QL system call summary, and QDOS error codes.

It is nothing if not comprehensive, but I was not exactly bowled over by the book's presentation. After the bold text of the contents page, the cramped, apparently typewritten text of the rest of the book was most disconcerting, especially since many pages have not reproduced very clearly. This comes across as a pointless piece of cost-cutting.

Fun And Games

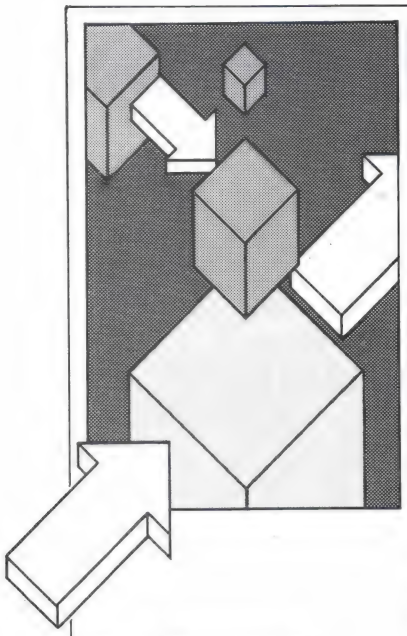
And so to this month's games corner. You may have purchased your QL for good, solid businesslike reasons, but as Tim Hartnell hastens to point out in his *QL Games Compendium*, Interface Publications there's absolutely no reason why you shouldn't play games as well on it, and have some fun. (Well, he would say that, wouldn't he.)

So what do you get for your £5.95? You get 23 games listings, grouped into seven sections: Adventure (including Tryst with Memphistopehles); Artificial Intelligence (including the famous Eliza program: Giggle Palace ("games designed just for fun" says the introduction – so what are the rest of them for?); Simulations (Stockbroker and Life); Board Games; games using moving graphics; and Deduction and Perception.

All of which has a distinctly familiar sound. Still, some of these old favourites will always be fun, like Eliza and Life, and when you've got bored with playing games, you can always amuse yourself wondering what on earth the glossary of computer terms at the back is for . . .

Finally, *The QL Book of Games* by Richard Hurley and David Virgo and published by Micro Press offers 13 games for £6.95 (compared to the 24 provided in Tim Hartnell's Compendium), but admittedly there's a more polished look to this book, and the listings tend to be longer.

Again, the games content lacks originality – there's horse racing, Othello, backgrammon, Sub Hunt, Tank Attack, Nightmare Park . . . you know. Each listing is introduced with a refreshing lack of pretension – a brief description, a few words about the program, a few lines of playing instructions, and straight into the program. There's also a few pages at the back enlarging on some of the BASIC keywords used.



THE PROGS

This is the place to look for readers' QL programs. So, if you've got a computational masterpiece, why not send it in for evaluation. The address is 'The Progs', QL User, Priory Court, 30-32 Farringdon Lane, EC1R 3AU. We pay for everything published.

Do It Yourself Adventure

P J Bagnell Smith

What is the point of typing-in a BASIC adventure? Certainly by the time you have entered it you will know every way to cheat and all the locations. For this reason we have decided to break with convention and publish an adventure with a difference. This one does not work!

The reason for this is that the program is a generalised skeleton for an adventure game. It is designed in such a fashion that you 'fill-in' the adventure proper and it takes care of the technicalities. The object is to remove the drudgery and leave you free to exercise your imagination and create your own game with a minimum of effort.

Getting down to the 'nitty-gritty' there are 10 places in the program where you must 'fill-in-the-gaps'. Before explaining where these are we suggest that you type-in (and save) the program so as to familiarise yourself with its structure. Once this is done we can move on . . . in order of appearance, the gaps to be filled relate to:

i) The number of rooms.
This is done by changing the variable in line 260 (no_of_ rooms) to the desired value. Remember to count all the locations that will be used.

ii) The number of monsters/witches etc . . .
This is done by changing the variable no_of_mons in line 300.

iii) The number of objects in the adventure.
This variable is called "no_of_ obs" and is situated in line 340.

iv) The instructions.
There must be instructions in the adventure. They go after line 690. Hint: use this line when you want to wait for any key to be pressed: –
<<line no.>> REPeat loop:IF INKEY$<>'' THEN EXIT loop

v) Data for rooms.
The data is held in this format:
<<line number>> DATA 3,2,7,8

This means that if you go from that room north you go to room 3, south 2, east 7 and west 8. Four elements are needed for each room and they are held in data statements from line 1000 onwards. So if you have 40 rooms you will require 160 items of data whose values range from 1 to 40 inclusive.

vi) Data for objects.
As above but from line 1040 in the format: –
<<line number>> DATA 'Vase',32,1

This means that the object is a vase, is at room 32, is mobile, and can be taken. If the final parameter was 0, then the vase could not be moved. So if you have ten objects you should have 30 data elements arranged in the correct format.

vii) Data for monsters.
As above but in the form: –
<<line number>> DATA 'Hag',21

This shows there is a hag at location no. 21 (not a pleasant event at all!)

viii) Data for rooms.
Here entries do not take the form of elements in a DATA statement but are conditions and statements on a SELect construction located at line 1520. Entries take the following form:
<<line number>> = 23
<<line number>> if location(1) then print 'You see the hag.' :else print 'You see the dead hag.'

A simple example would be:
<<line number>> = 1
<<line number>> PRINT 'The great lake.'
(ie – when you are at location 1 the computer will tell you you are at the great lake, there is no need to give information on the objects in the location.)

ix) Object Usage: Combinations.
Here insertions into the procedure called "use" at line 2650 determine which objects may be used at what location. This information is entered in the form: –
<<line number>> if ql = 23 and room = 15 then flag = 1

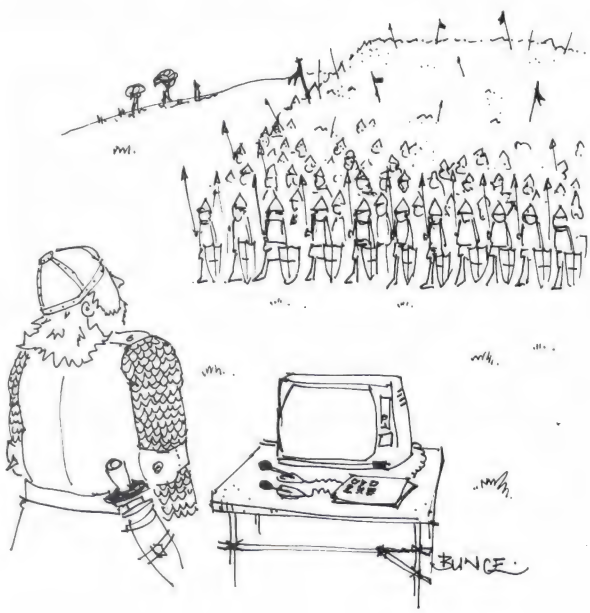
This means that if you are trying to use object 23 in room number 15 then that combination is acceptable.

x) Object Usage: Consequences.
This section of the procedure (line 2690) must be totally rewritten for each new adventure.

The commands must be in the form:
<<line number>> if ql = 23 and room = 15 then print 'The key unlocks and opens the door.' :direction (room, east) = 23
(ie – if the key is being used and you are in room 15 then the door is unlocked and if you then go east you go to room number 23.)

Variables List
If the adventure is to work the variables listed below should be used in particular when 'filling-in' the consequences section.

North = 1 This is always set to 1, to help you remember the order of the parameters in the data statements (see the 10th thing to change's example), the same is true of the next 3 variables,
south = 2
east = 3
west = 4
no Number of objects held,
hold$() The objects held in the order they were picked up in, so if for example you open a door you get a door knob (ob 14), this could be used: –
<<line number>> . . . . . no = no + 1:hold$(no) = 'Door knob' :object (14,1) = 0:print 'The door knob falls off in your hand as you open the door.'
direction (room,direction)
Has two parameters: –
i) room number,
ii) direction you wish to go in, (uses the north = 1, south = 2 convention outlined above)
This returns the room you will go to, ie,
if direction (12,north) = 23 then print 'From room 12, if you go north you will go to room 23.'
object (number,1)
This is the room that the object is in, see hold$ above. If the object is held then the equation equals 0. See object$, below.
object (number,2)
This shows if the object is mobile, (ie the object can be taken), 1 = mobile, 0 =



Send forth your champion!

immobile, this can be shown this way; – If object (23,1) then print 'You can take the dagger.'

object$ (object) This is the name of the object: ie, <<line number>> If object$ (1) = 'Vase' then print 'Object 1 is a vase.' or, <<line number>> if object$ (1,1) = 0 then print 'You have the vase.'

monster$ (monster) This is the name of the monster ie, <<line number>> if monster$ (1) = 'Hag' and location (1) = room then print 'The hag is in the same room as you are.'

location (monster) This is the room the monster is in, the only useful purpose it has is shown below . . . <<line number>> if ql = 2 and room = location (2) then print 'You killed the monster.' :location (2) = 0

room This is the number of the room you are in, uses for this are shown above.

flags This will be of no interest to you, but it tells the program if a statement has been understood.

flag This is used many times to tell the program if a condition is true. For example if you try to use a rock, does one exist?

ql This is the value of the object you are trying to use. ie, <<line number>> Print 'You try to use the' ;object$ (ql);' but it breaks.'

When you have finished the program remember to IMPORT the program onto Quill and dispose of all occurrences of "no_of_rooms", no_of_mons" and "no_of_obs" and replace them with their values. Also as you type in long data lists, repeatedly RENUMber, bearing in mind that some of the line numbers shown above will no longer relate to the adventure, if you keep the REM statements in the program it will be easier to find the desired piece of program to hack at.

```
110 REMark DIY Adventure
120 REMark for QL User
130 REMark 1984  Bagnell Smith
140 REMark

160 :
170 :
180 :
190 REPeat loop
200 set_up_screen
210 instructions
220 north=1
230 south=2
240 east=3
250 west=4
260 no_of_rooms=40
270 REMark ****************************************
*****
280 REMark * Set this variable to the number of ro
oms *
290 REMark ****************************************
*****
300 no_of_mons=2
310 REMark ****************************************
320 REMark * Set this to the number of monsters *
330 REMark ****************************************
340 no_of_obs=17
350 REMark ****************************************
360 REMark * Set this to the number of objects *
370 REMark ****************************************
390 RESTORE
400 set_up_variables
410 REPeat no_exit
420 play_the_game
440 END REPeat no_exit
450 INPUT \\\'Press ENTER to re-play.';enter$
460 END REPeat loop
470 :
480 REMark END OF PROGRAM LOOP
490 :
500 DEFine PROCedure set_up_screen
510 MODE 4
520 OPEN#3,scr_512x256a0x0
530 PAPER#3,2
540 CLS#3
550 OPEN #1,con_384x200a64x6
560 CSIZE 0,0
570 PAPER 0
580 INK 7
590 CLS
600 PAPER#0,2
610 INK#0,7
620 OPEN #2,con_384x200a64x6
630 END DEFine
640 :
650 :
660 :
670 DEFine PROCedure instructions
680 CLS
690 REMark ****************************
700 REMark * The instructions go here *
710 REMark ****************************
720 END DEFine
730 :
740 :
750 :
760 DEFine PROCedure set_up_variables
770 no=0
780 DIM hold$(no_of_obs,10)
790 DIM direction(no_of_rooms,4)
800 FOR room=1 TO no_of_rooms:FOR di=1 TO 4:READ d
irection(room,di):NEXT di:NEXT room
810 DIM object(no_of_obs,2)
820 DIM object$(no_of_obs,10)
830 FOR thing=1 TO no_of_obs
840 READ object$(thing)
850 READ object(thing,1)
860 READ object(thing,2)
870 END FOR thing
880 DIM monster$(no_of_mons,7)
890 DIM location(no_of_mons)
900 FOR monster=1 TO no_of_mons
910 READ monster$(monster)
920 READ location(monster)
930 END FOR monster
940 room=1
950 END DEFine
960 :
970 :
980 :
990 REMark ************************************
1000 REMark * Data for rooms in format N,S,E,W *
1010 REMark ************************************
1020 :
1030 REMark ****************************************
*************
1040 REMark * Data for objects in format name$,roo
m,moveable? *
1050 REMark ****************************************
*************
1060 :
1070 REMark ****************************************
****
1080 REMark * Data for monsters if format name$,ro
om *
1090 REMark ****************************************
****
1100 :
1110 :
1120 :
1130 DEFine PROCedure play_the_game
1140 flags=0
1150 print_monster
1160 print_location
```

```
1170 print_object
1180 print_directions
1190 INPUT 'Your command master...';command$
1200 IF 'N' INSTR command$=1 THEN goo(north):flags
=1
1210 IF 'S' INSTR command$=1 THEN goo(south):flags
=1
1220 IF 'E' INSTR command$=1 THEN goo(east):flags=
1
1230 IF 'W' INSTR command$=1 THEN goo(west):flags=
1
1240 IF 'inst' INSTR command$=1 THEN instructions:
flags=1
1250 IF 'help' INSTR command$=1 THEN PRINT 'What a
 shame !!!!':flags=1
1260 IF 'take' INSTR command$=1 THEN take:flags=1
1270 IF 'drop' INSTR command$=1 THEN drop:flags=1
1280 IF 'use' INSTR command$=1 THEN use:flags=1
1290 IF 'inv' INSTR command$=1 THEN inventory:flag
s=1
1300 IF 'kill' INSTR command$=1 THEN kill:flags=1
1310 IF 'quit' INSTR command$=1 THEN exit_flag=1:R
ETurn
1320 IF NOT flags THEN PRINT \'Statement not under
stood.'\'Please re-phrase.'\\
1330 END DEFine
1340 :
1350 :
1360 :
1370 DEFine PROCedure print_monster
1380 FOR qwerty=1 TO no_of_mons
1390 IF room=location(qwerty) THEN PRINT 'There is
 a ';monster$(qwerty)
1400 END FOR qwerty
1410 END DEFine
1420 :
1430 :
1440 :
1450 DEFine PROCedure print_location
1460 SELect ON room
1470 :
1480 REMark * data in format
1490 REMark * =room_number
1500 REMark * PRINT discription$
1510 :
1520 END SELect
1530 END DEFine
1540 :
1550 :
1560 :
1570 DEFine PROCedure print_object
1580 flag=0
1590 PRINT 'You see:'
1600 FOR qwerty=1 TO no_of_obs
1610 IF object(qwerty,1)=room THEN PRINT ' ';objec
t$(qwerty):flag=1
1620 END FOR qwerty
1630 IF NOT flag THEN PRINT ' Nothing.'
1640 PRINT
1650 END DEFine
1660 :
1670 :
1680 :
1690 DEFine PROCedure goo(x)
1700 IF direction(room,x)=0 THEN PRINT 'You are un
-able to go that way.':RETurn
1710 room=direction(room,x)
1720 END DEFine
1730 :
1740 :
1750 :
1760 DEFine PROCedure print_directions
1770 PRINT 'You can go:'
1780 flag=0
1790 IF direction(room,north) THEN PRINT ' North.'
:flag=1
1800 IF direction(room,south) THEN PRINT ' South.'
:flag=1
1810 IF direction(room,east) THEN PRINT ' East.':f
lag=1
1820 IF direction(room,west) THEN PRINT ' West.':f
lag=1
1830 IF flag=0 THEN PRINT ' No where.'
1840 PRINT
1850 END DEFine
1860 :
1870 :
1880 :
1890 DEFine PROCedure take
1900 flag=0
1910 FOR qwerty=1 TO no_of_obs
1920 IF object(qwerty,1)=room THEN flag=1
1930 END FOR qwerty
1940 IF NOT flag THEN PRINT 'No object.':RETurn
1950 flag=0
1960 FOR qwerty=1 TO no_of_obs
1970 IF object(qwerty,1)=room AND object(qwerty,2)
=1 THEN flag=1
1980 END FOR qwerty
1990 IF NOT flag THEN PRINT 'The object is im-mobi
le.':RETurn
2000 IF LEN(command$)<6 THEN INPUT 'Take what...';
command$:command$='Take '&command$
2010 command$=command$(6 TO)
2020 flag=0
2030 FOR qwerty=1 TO no_of_obs
2040 IF command$ INSTR object$(qwerty)=1 AND objec
t(qwerty,1)=room AND object(qwerty,2) THEN no=no+1
:hold$(no)=object$(qwerty):object(qwerty,1)=0:flag
=1
2050 END FOR qwerty
2060 IF NOT flag THEN PRINT 'What ?':RETurn
2070 PRINT 'You have The ';hold$(no)
2080 END DEFine
2090 :
2100 :
2110 :
2120 DEFine PROCedure drop
2130 IF no=0 THEN PRINT 'You do not have any-thing
.':RETurn
2140 IF LEN(command$)<6 THEN INPUT 'Drop what...';
command$:command$='Drop '&command$
2150 command$=command$(6 TO)
2160 flag=0
2170 FOR qwerty=1 TO no_of_obs
2180 IF command$ INSTR object$(qwerty)=1 AND objec
t(qwerty,1)=0 THEN flag=1
2190 END FOR qwerty
2200 IF NOT flag THEN PRINT 'You do not have it.':
RETurn
2210 FOR qwerty=1 TO no_of_obs
2220 IF command$ INSTR object$(qwerty)=1 AND objec
t(qwerty,1)=0 THEN object(qwerty,1)=room:EXIT qwer
ty
2230 END FOR qwerty
2240 FOR q=1 TO no
2250 IF hold$(q)=object$(qwerty) THEN EXIT q
2260 END FOR q
2270 hold$(q)=''
2280 FOR w=q TO no-1
2290 hold$(w)=hold$(w+1)
2300 END FOR w
2310 PRINT 'It is dropped.'
2320 no=no-1
2330 END DEFine
2340 :
2350 :
2360 :
2370 DEFine PROCedure inventory
2380 PRINT 'You have...'
2390 FOR qwerty=1 TO no
2400 PRINT ' ';hold$(qwerty)
```

```
2410 END FOR qwerty
2420 IF no=0 THEN PRINT ' Nothing.'
2430 PRINT
2440 END DEFine
2450 :
2460 :
2470 :
2480 DEFine PROCedure use
2490 flag=0
2500 IF LEN(command$)<5 THEN INPUT 'Use what...';c
ommand$:command$='Use '&command$
2510 command$=command$(5 TO)
2520 FOR qwerty=1 TO no_of_obs
2530 IF command$ INSTR object$(qwerty)=1 THEN flag
=1:ql=qwerty
2540 END FOR qwerty
2550 IF NOT flag THEN PRINT 'That does not even ex
ist!':RETurn
2560 flag=0
2570 FOR qwerty=1 TO no
2580 IF command$ INSTR hold$(qwerty)=1 THEN flag=1
2590 END FOR qwerty
2600 IF NOT flag THEN PRINT 'You do not have that.
':RETurn
2610 flag=0
2620 :
2630 :
2640 :
2650 REMark * Lines in format
2660 REMark * IF QL=(number of object) AND room=(n
umber of room that object can be used in) then fla
g=1
2670 IF NOT flag THEN PRINT "You can't use that 'e
re!":RETurn
2680 REMark ****************************************
****
2690 REMark * This section must be totaly re-writt
en *
2700 REMark ****************************************
****
2710 END DEFine
2720 :
2730 :
2740 :
2750 DEFine PROCedure kill
2760 IF LEN(command$)<6 THEN INPUT 'Kill what...';
command$:command$='Kill '&command$
2770 command$=command$(6 TO)
2780 flag=0
2790 FOR qwerty=1 TO no_of_mons
2800 IF command$ INSTR monster$(qwerty)=1 THEN fla
g=1:ql=qwerty
2810 END FOR qwerty
2820 IF NOT flag THEN PRINT 'It does not exist.':R
ETurn
2830 IF NOT location(ql) THEN PRINT 'It is allread
y dead.':RETurn
2840 qwerty=RND
2850 IF qwerty>=.5 THEN PRINT 'It is dead!!':locat
ion(ql)=0:RETurn
2860 PRINT 'It killed YOU.'
2870 EXIT no_exit
2880 END DEFine
2890 :
2900 :
2910 :
2920 REMark
                                        The END.
2930 :
2940 :
2950 :
3000 DEFine PROCedure cat(a)
3005 CLS
3010 DIR 'mdv'&a&'_'
3030 END DEFine
3040 :
```

The Missing Link

In our letters section, a reader has written in suggesting that a list of QDOS vector and subroutine addresses be published. He also adds that without subroutines for keyboard input and screen output an assembler is next to useless. How right he is!

For this very reason four separate publishers have already brought out comprehensive manuals on the subject. However, if you haven't got the relevant documentation at hand then the following list of the more commonly used addresses should prove invaluable.

If you have an assembler that supports macro commands (ie Metacomco or GST) you need never re-enter these codes, simply save this file as you would a text file for compilation and then use the GET or INCLUDE directives in your assembler listing. ie **GET "mdvt_header_asm"**

Finally, those in a rush to put this listing to use need look no further than our previous issue where both multi-tasking programs incorporated a similar file (which incidentally, had been omitted).

```
* Operating system vectors

UT_WINDW  EQU     $C4           Opens a window using a supplied name
UT_CON    EQU     $C6           Opens a pre-defined console dev.
UT_SCR    EQU     $C8           Opens a pre-defined screen device
UT_ERRO   EQU     $CA           Prints error message to channel 0
UT_ERR    EQU     $CC           Prints error message to channel
UT_MINT   EQU     $CE           Prints an integer to a channel
UT_MTEXT  EQU     $D0           Prints a string to a channel
UT_CSTR   EQU     $E6           Compares two string
CN_DATE   EQU     $EC           Converts D1 clock to ASCII
CN_DAY    EQU     $EE           Converts D1 day to ASCII
CN_FTOD   EQU     $F0           Converts floating point to ASCII
CN_ITOD   EQU     $F2           Converts integer to ASCII
CN_ITOBB  EQU     $F4           Converts a byte to ASCII binary
CN_ITOBW  EQU     $F6           Converts a word to ASCII binary
CN_ITOBL  EQU     $F8           Converts a long word to ASCII binary
CN_ITOHB  EQU     $FA           Converts a byte to ASCII hex
CN_ITOHW  EQU     $FC           Converts a word to ASCII hex
CN_ITOHL  EQU     $FE           Converts a long word to ASCII hex
CN_DTOF   EQU     $100          Converts ASCII to floating point
CN_DTOI   EQU     $102          Converts ASCII to integer
BP_INIT   EQU     $110          Routine to link procedures
CA_GTINT  EQU     $112          Gets arguments as integers
CA_GTFP   EQU     $114          Gets arguments as floats
CA_GTSTR  EQU     $116          Gets arguments as strings
CA_GTLIN  EQU     $118          Gets arguments as long integers
BV_CHRIX  EQU     $11A          Reserves D1.L RI satck space
RI_EXEC   EQU     $11C          Performs various RI arith ops
RI_EXECB  EQU     $11E          Performs a block of RI arith ops
BP_LET    EQU     $120          Performs a variable assignment

* Operating system offsets and equates

SV_BASE   EQU     $28000        Start of system variables
CH_LENCH  EQU     $28           Length of Basic channel entry
BV_CHBAS  EQU     $30           Offset from A6 of Basic channels
BV_RIP    EQU     $58           Offset from A6 of RI SP

ERR_NC    EQU     -1            'Not complete' error
ERR_NJ    EQU     -2            'Invalid job' error
ERR_OM    EQU     -3            'Out of memory' error
ERR_OR    EQU     -4            'Out of range' error
ERR_BO    EQU     -5            'Buffer overflow' error
ERR_NO    EQU     -6            'Not open' error
ERR_NF    EQU     -7            'Not found' error
```

```
ERR_EX    EQU     -8              'Already exists' error
ERR_IU    EQU     -9              'In use' error
ERR_EF    EQU     -10             'End of file' error
ERR_DF    EQU     -11             'Drive full' error
ERR_BN    EQU     -12             'Bad name' error
ERR_TE    EQU     -13             'Transmission error' error
ERR_FF    EQU     -14             'Format failed' error
ERR_BP    EQU     -15             'Bad Parameter' error
ERR_FE    EQU     -16             'Bad or changed medium' error
ERR_XP    EQU     -17             'Error in expression' error
ERR_OV    EQU     -18             'Overflow' error
ERR_NI    EQU     -19             'Not implemented' error
ERR_RO    EQU     -20             'Read only' error
ERR_BL    EQU     -21             'Bad Line' error

RET_STR   EQU     1               Function return type string
RET_FP    EQU     2               Function return type float
RET_INT   EQU     3               Function return type integer

OPEN_INX  EQU     0               Open old exclusive
OPEN_INS  EQU     1               Open old shared
OPEN_NEW  EQU     2               Open new exclusive
OPEN_OVR  EQU     3               Open overwrite
OPEN_DIR  EQU     4               Open directory

* Trap keys
* Trap 1

MT_CJOB   EQU     1               Create a job
MT_JINF   EQU     2               Information on job tree
MT_FRJOB  EQU     5               Force release a job
MT_TRAPV  EQU     7               Set the exception vectors for a job
MT_SUSJB  EQU     8               Suspend a job
MT_RELJB  EQU     9               Release a job
MT_ACTIV  EQU     $A              Activate a job
MT_PRIOR  EQU     $B              Change job priority
MT_ALRES  EQU     $E              Allocate resident procedure area
MT_RERES  EQU     $F              Release resident procedure area
MT_DMODE  EQU     $10             Read/set the display mode
MT_IPCOM  EQU     $11             Send command to 8049 IPC
MT_RCLCK  EQU     $13             Read the clock into D1.L
MT_SCLCK  EQU     $14             Set the clock from D1.L
MT_ACLCK  EQU     $15             Adjust the clock from D1.L
MT_ALCHP  EQU     $18             Allocate common heap area
MT_RECHP  EQU     $19             Release common heap area
MT_LXINT  EQU     $1A             Link in external interrupt handler
MT_RXINT  EQU     $1B             Remove external interrupt handler
MT_LPOLL  EQU     $1C             Link in polled task
MT_RPOLL  EQU     $1D             Remove polled task
MT_LSCHD  EQU     $1E             Link in scheduler task
MT_RSCHD  EQU     $1F             Remove scheduler task
MT_LIOD   EQU     $20             Link in I/O driver
MT_RIOD   EQU     $21             Remove I/O driver
MT_LDD    EQU     $22             Link in directory device driver
MT_RDD    EQU     $23             Remove directory device driver

* Trap 2

IO_OPEN   EQU     1               Open a channel
IO_CLOSE  EQU     2               Close a channel
IO_FORMT  EQU     3               Format a medium
IO_DELET  EQU     4               Delete a file

* Trap 3

IO_PEND   EQU     0               Check channel for pending input
IO_FBYTE  EQU     1               Fetch a byte from a channel
IO_FLINE  EQU     2               Fetch a line
IO_FSTRG  EQU     3               Fetch a string of bytes
IO_EDLIN  EQU     4               Edit a console line
IO_SBYTE  EQU     5               Send a byte to a channel
IO_SSTRG  EQU     7               Send a string to channel
SD_WDEF   EQU     $D              Redefine a window
SD_CURS   EQU     $F              Disable the cursor on a channel
SD_POS    EQU     $10             Set cursor position absolute
SD_TAB    EQU     $11             Tab cursor in channel
SD_NCOL   EQU     $14             Move cursor to next column
SD_CLEAR  EQU     $20             Clear entire window
SD_FOUNT  EQU     $25             Alter fount address
SD_POINT  EQU     $30             Plot a point
SD_LINE   EQU     $31             Draw a line
SD_ARC    EQU     $32             Draw an arc
SD_ELIPS  EQU     $33             Draw a circle or ellipse
SD_SCALE  EQU     $34             Change the graphics scale
SD_GCUR   EQU     $36             Position the graphics cursor
FS_CHECK  EQU     $40             Check a file's pending operations
FS_FLUSH  EQU     $41             Flush a file's buffers
FS_POSAB  EQU     $42             Set pointer absolute
FS_POSRE  EQU     $43             Set pointer relative
FS_MDINF  EQU     $45             Read medium information
FS_HEADS  EQU     $46             Write file header
FS_HEADR  EQU     $47             Read file header
FS_LOAD   EQU     $48             Load a complete file
FS_SAVE   EQU     $49             Save a complete file

* RI operation keys

RI_FLOAT  EQU     8               Integer to fp TOS
RI_ADD    EQU     $A              TOS fp + NOS fp --> TOS
RI_MULT   EQU     $E              TOS fp * NOS fp --> TOS
```

Pseudo Editor

The absence of a full screen editor, by now standard, even on cheap 8-bit home computers comes as a considerable blow bearing in mind SuperBasic's advanced features. An efficient and easy to use editor is vital if one is to get the most out of programming.

Unlike editors for compiled languages, a BASIC editor does not only format text but also debugs program lines as they are entered. As such it is a mini-interpreter in its own right and as a consequence the design is far from simple and well beyond the scope of BASIC.

Given these constraints, the following compromise written wholly in SuperBasic is quite ingenious. The author has put together a series of short routines that occupy the front end of a program. These routines remain invisible throughout, permitting one to scroll or page backwards and forwards through a listing using the cursor keys.

Whilst not independent of the line editor, the program enables one to switch between "Edit" and "Display" modes with consummate ease. In "Edit" mode, amendments or insertions are typed-in using the QL's line editor and when finished, entering "RUN" will return one to "Display" mode, with the cursor keys once again re-enabled.

As after an insertion or a deletion the program automatically RENUMbers your program. PROCedures should be used instead of GOSUB's and GOTO's should be scrupulously avoided. As this encourages structured programming it can only be considered a benefit.

Following Psion's example, the author has also included a "HELP" function (F1) that makes any detailed explanation of operating procedures virtually redundant.

Having entered the

THE PROGS

program, users are advised to save the program prior to running it. The reason for this, is that line 6 renumbers itself as line 5 after the program has been run for the first time so that the title sequence is not repeated.

Finally, when merging programs into this editor note that they must not contain lines whose numbers are less than 100, otherwise when merged the editor itself will be overwritten. With this single proviso, the program should prove indispensable to the basic programmer fed up with having to list chunks of his programs on the screen every time he wishes to amend them.

```
 1  REMark PROGRAM LISTING 1.1 - M. D. Newport, 19
84
2  WINDOW 448,170,32,30: WINDOW #0, 448,50,32,206
3  OPEN #3, con_448x10a32x16
4  n=6: GO TO n
6  CLOSE #2: ch2$="closed": lne=100: title: RENUM
6 TO 7;5,1
7  RENUM 100 TO 32760;100,10
8  IF ch2$="open": CLOSE #2: ch2$="closed"
9  MODE 4: display
10 REPeat choice
11    key=CODE(INKEY$)
12    SELect ON key
13       =216: lne=lne+10: scroll_down
14       =208: lne=lne-10: scroll_up
15       =200: lne=lne+100: display
16       =192: lne=lne-100: display
17       = 32: jump: display
18       = 10: mode_type 2: EXIT choice
19       =232: help: display
20       =240: run_program
21       =236: save_quit: EXIT choice
22    END SELect
23 END REPeat choice
24 STOP:  REMark _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _ _
25 DEFine PROCedure display
26    BORDER 5,4: PAPER 7: INK 2: CSIZE 0,0: CLS: m
ode_type 1
27    IF lne<100: lne=100
28    FOR i= 1 TO 14: AT i,0: LIST #1, lne+10*(i-1)
29    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
30 DEFine PROCedure scroll_down
31    AT 2,0: CLS 1: SCROLL -10: AT 14,0: LIST #1,
lne+130
32    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
33 DEFine PROCedure scroll_up
34    IF lne=90: lne=lne+10: RETurn
35    AT 13,0: CLS 2: SCROLL +20
36    AT 1,0: LIST #1, lne: AT 1,0: SCROLL -10,2
37    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
38 DEFine PROCedure jump
39    WINDOW #3, 184,10,32,16: CLS #3
40    INPUT #3, "  START AT LINE NUMBER: "; k
41    lne=10*INT((k+9)/10)
42    PAPER #3, 0: CLS #3
43    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
44 DEFine PROCedure mode_type(t)
45    WINDOW #3, 84,10,396,16: PAPER #3, 4: INK #3,
 0
46    SELect ON t
47       =1: PRINT #3, " DISPLAY MODE ": RETurn
48       =2: PRINT #3, "   EDIT MODE  ": RETurn
49       =3: PRINT #3, "   HELP MODE  ": RETurn
50    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
51 DEFine PROCedure help
52    mode_type 3: PAPER 4: CLS: INK 0
53    FOR y=10, 34, 72, 92: LINE 0,y TO 200,y
54    LINE 27,0 TO 27,100: LINE 130,34 TO 130,72: I
NK 2
55    AT 0,0: PRINT "  MODE ": AT 0,33: PRINT " COM
MANDS "
56    AT 2,0: PRINT "  LOAD ": AT 7,0: PRINT " DISP
LAY "
57    AT 12,0: PRINT "  EDIT ": AT 15,0: PRINT "  H
ELP "
58    INK 0: AT 2,13: PRINT "Use  MRUN  from edit m
ode to load program to be listed."
59    AT 3,22: PRINT "If necessary  RENUM  before l
oading."
60    AT 5,11: PRINT "arrow right .. Page forward"
61    AT 5,47: PRINT " arrow left .. Page back"
62    AT 6,11: PRINT "arrow down ... Scroll down"
63    AT 6,47: PRINT " arrow up .... Scroll up"
64    AT 7,11: PRINT "space bar .. Jump to selected
 line": AT 7,51: PRINT " F1 ..... Help"
65    AT 8,11: PRINT " ENTER ..... Change to edit m
ode": AT 8,51: PRINT " F2 ..... Save/Quit"
66    AT 9,11: PRINT " F3  ......  Run listed progr
am"
67    AT 11,11: PRINT "All Superbasic editing facil
ities available:  EDIT  using"
68    AT 12,11: PRINT "cursor + CTRL  keys or type
in line.  Also  DLINE ,  RENUM ."
69    AT 13,25: PRINT " RUN .... Return to display"
70    AT 15,24: PRINT "space bar .... Return to dis
play"
71    WINDOW #3, 186,10,32,16: c=1
72    REPeat loop
73       IF c=6: PRINT #3, "  PRESS SPACE BAR TO CON
TINUE  "
74       IF INKEY$(0)=CHR$(32): EXIT loop
75       IF c=25: CLS #3: c=0
76       c=c+1
77    END REPeat loop
78    PAPER #3, 0: CLS #3
79    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
80 DEFine PROCedure save_quit
81    PAPER #3, 0: CLS #3
82    WINDOW 280,70,116,46: BORDER 2,2: PAPER 0: IN
K 7: CLS
83    AT 1,6: PRINT "Save listed program to microdr
ive"
84    AT 2,2: PRINT "e.g. SAVE MDV1_program_name, 1
00 to 32760"
85    AT 4,12: PRINT "Then RUN to continue"
86    AT 5,6: PRINT "or RESET to return to Superbas
ic."
87    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
88 DEFine PROCedure run_program
89    PAPER #3, 0: CLS #3: CLOSE #3: CLS #0: CLS
90    OPEN #2,con_448x170a32x30: ch2$="open": RUN 1
00: RETurn
91    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
92 DEFine PROCedure title
93    MODE 4: BORDER 5,4: PAPER 7: CLS
94    BLOCK 274,40,70,30,0: BLOCK 268,36,72,32,2
95    INK 0: CSIZE 3,1: AT 2,5: PRINT "PROGRAM  LIS
TING"
96    CSIZE 2,0: AT 10,7: PRINT "M.D.Newport      19
84"
97    CSIZE 1,0: AT 13,14: PRINT "( For QL User Mag
azine )"
98    PAUSE 130: CSIZE 0,0: help
99    END DEFine        : REMark _ _ _ _ _ _ _ _ _
_ _ _ _ _ _ _ _ _ _ _ _ _
32766 REMark This line prevents line 32767 being r
enumbered.
32767 STOP
```

CLASSIFIED

QL PAYROLL

★ WEEKLY, MONTHLY OR 4 WEEKLY ★ TAX, NI, P60, P35 CALCULATION
★ USER DEFINABLE DEDUCTIONS ★ COIN ANALYSIS/PAYMENT ROUNDING
★ SUPPORT COVER AVAILABLE

£55 including postage and packaging. — (Please add 15% VAT)

Further details of this and other packages for Stock, Accounts, etc. are available from:

TR COMPUTER SYSTEMS
HEATHFIELD, HINE HEATH, STANTON, NR. SHREWSBURY, SALOP SY4 4NA
Telephone: (093 924) 621

QL BARGAINS

Prices include VAT and carriage

QL COMPUTER + PSION CHESS — £399

Psion Chess	£16	4 Microdrive carts.	£19
10 Microdrive cartridges + Transform box for 20			£49
Serial Cable	£11	Parallel Interface	£35

PRINTERS

Brother M1009	£178	Seikosha GP550A	£199
Shinwa CPA80P	£219	Mannesmann Tally MT80	£219
Epson RX80FT	£254	Canon PW1080A	£329
Smith Corona TP-1	£209	Daisy Step 2000	£299
One only Seikosha GP700A Colour Ex-demo full warranty			£275

MONITORS

Microvitec Cub	£259	Prism QL14	£199
Philips Hi-Res. Mono Chrome			£87

STRONG COMPUTER SYSTEMS
Bryn Cottage, Peniel, Carmarthen, Dyfed SA32 7DS
Telephone: (0267) 231246 for Assistance!!!

★ ★ TOP TEAM ★ ★

Football management game of strategy and skill!

Choose your team from 24 first and second division clubs, then use your skill to manage the team through the 1990 Top Team Championship.
Features: Tactic selection, named players, league tables, scouts reports and real-time text commentary on your games.

£7.95 (Add £5 for cartridge or send you own).

Send cheque/P.O. made payable to **ARUNSOFT** –
6 The Meadows, Walberton, Nr. Arundel, W. Sussex BN18 0PB
(We will pay good royalties for your quality original programs).

QL UTILITIES

FOUR Programs on Microdrive for Sinclair QL to prevent Directory overflowing the screen, provide single-key LOADing or DELETion of files, repeat FORMATing of cartridges and back-up COPYing of whole or part of cartridge.

£10 from **WDSoftware**, Hilltop, St Mary, Jersey, C.I.
Tel: (0534) 81392

QL FOR BUSINESS
OWL Computer Training

A programme of linked courses.
Introductory & Advanced Applications.

044 282 7302
FREEPOST, TRING, HERTS. HP23 4BR.

EXECUTIVE ADVENTURE

Can YOU rise from lowly tramp to high flying executive, in this unusual modern day adventure. Pit your wits against puzzles, logical problems, mazes; test your skills in money management and devious dealings. No dragons in this original adventure (except the secretary)!

£15.00

INTERSOFT, 7 Richmond Road, Exeter

QLPRINT

Your QL documents/listings printed £2 per file any length inc. p&p. Send your tape and cheque or PO to

QLPRINT, Chubbs Cottage, Buddle Lane, Hatherliegh, Devon EX20 3HX or phone Hatherliegh 535

ADVERTISERS INDEX

QL POWER

AT YOUR FINGERTIPS FROM COMPUTER ONE

PASCAL £39.95 Inc. VAT & p&p.

Computer One PASCAL is a powerful implementation of this classic programming language. Produced specifically for the QL, this package provides a professional and highly educational programming system. Complete with comprehensive 80 page manual, this is the ultimate language package that no QL enthusiast can do without.

INCLUDES: ★ Full Screen Editor ★ QL graphics and sound extensions
★ Example programmes ★ Complete 87 page User Guide
★ Full implementation close to ISO-standard ★ Typically 15x faster than BASIC

FORTH £39.95 Inc. VAT & p&p.

FORTH is easy to learn, highly efficient, and allows you to explore the full capabilities of your QL. Computer One FORTH is a full implementation of the latest FORTH-83 standard with graphics and sound extensions, and may be used to produce machine code applications for your QL. Ideal for writing real-time or games programs.

INCLUDES: ★ FORTH-83 system ★ Forth Screen Editor ★ 48-page manual
★ QL-graphics and sound extensions ★ Example FORTH programs

ASSEMBLER £29.95 Inc. VAT & p&p.

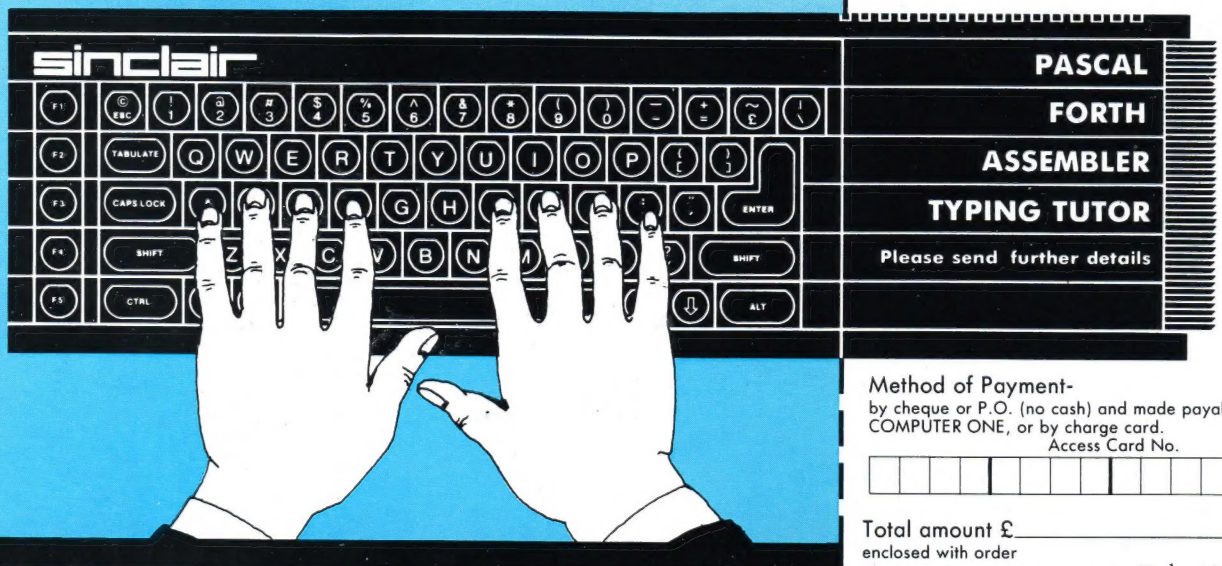
COMPUTER ONE Assembler is a comprehensive assembler program toolkit providing a professional multitasking assembler written in machine code for speed and compactness. An ideal package for the machine code programmer, allowing you easy access to the QDOS operating system, and integration to your Superbasic and Pascal programs.

INCLUDES: ★ 68008 Assembler ★ Full Syntax Checking ★ Full screen editor
★ Comprehensive User Manual ★ Integrates to Pascal and Superbasic

TYPING TUTOR £19.95 Inc. VAT & p&p.

Let your Sinclair QL teach you to type! Computer One's Typing Tutor is a fast, easy and enjoyable way of learning true touch-typing on your QL.

INCLUDES: ★ Self-paced tutorials
★ Speed and accuracy diagnosis
★ Interactive colour graphics
★ User Guide



computer ONE

SCIENCE PARK, MILTON ROAD,CAMBRIDGE

Name_____

Address_____

_____

Signed_____

Tick Order

PASCAL ☐
FORTH ☐
ASSEMBLER ☐
TYPING TUTOR ☐
Please send further details ☐

Access

Method of Payment-
by cheque or P.O. (no cash) and made payable to COMPUTER ONE, or by charge card.
Access Card No. QL/2/85

Total amount £_____
enclosed with order

Credit card sales: UK buyers only. Tel: (0223) 862616

The first QL adventures from TALENT!

The Lost Kingdom of ZKUL

Explore the ancient dwelling-place of the Dwarves – where the Wizard guards the last precious secret.

A classic and complex text adventure with hundreds of locations and a huge vocabulary. Set in real-time with traps, tasks, puzzles and mazes – and a special note-pad feature to aid you.

"the quality of the game is superb"
Micro Adventurer

PRICE
£19.95
+ 50p postage and packing

WEST

You are on the track of a notorious gang of bank robbers who have gone to ground near an abandoned mining town deep in Indian territory. Your task – to outwit and outgun the robbers, collect as much loot as possible and piece together clues on how to escape.

A demanding and exciting text adventure for the experienced games-player, using over 200 words and as many phrases. There are over 130 locations to search – not all of them easy to find. Events happen in real-time outside your control – Indians charge, rattle-snakes slither past and robbers appear and shoot at you. How long can you survive?

PRICE
£19.95
+ 50p postage and packing
(Also on CBM 64, BBC & Electron)

TALENT
COMPUTER SYSTEMS

CURRAN BUILDING, 101 ST JAMES ROAD, GLASGOW G4 0NS
24 HOUR CREDIT CARD HOT-LINE 041-552 2128